See 1473

(2)

October 1975

AD B008645

ATACM:
ACDA Tactical Air Campaign Model

ACDA/PAB-249

Prepared For

U.S. ARMS CONTROL
AND
DISARMAMENT AGENCY

Prepared By

KETRON, INC.

1400 Wilson Boulevard
Arlington, Virginia 22209

KFR No. 44-75

October 1975

ATACM:
ACDA Tactical Air Campaign Model

ACDA/PAB-249

John R. Fish

Prepared For

U.S. Arms Control and Disarmament Agency
21st and Virginia Avenues, N.W.
Washington, D.C. 20451

Prepared By

Ketron, Inc.
1400 Wilson Boulevard
Arlington, Virginia 22209

PAB-249

ABSTRACT

ATACM is a computer model designed and built for the Arms Control and Disarmament Agency for use in analyzing the impact of various force mixes upon a tactical airwar in Europe between NATO and Warsaw Pact forces. ATACM models an air campaign as a zero-sum staged game and employs dynamic programming to solve this game for approximate, optimal MAXMIN/MINMAX aircraft allocation strategies for the opposing sides at each stage of the campaign. The model permits multiple aircraft types with user-assigned missions, numerical and fractional reinforcements as a function of stage, and user selection of the objective function used to generate the optimal strategies.

Descriptions of the problem formulation and the engagement and optimization methodologies used to solve it are presented along with a user's guide and CDC 6600 FORTRAN listings.

PAB-249

ACKNOWLEDGEMENT

Many of the ideas presented here are the result of survey work by Drs. Fred Miercort and Robert Galiano and the author gratefully acknowledges their assistance. In addition, Captain Bernard M. Robinson, USAF, and Dr. George Pitman of ACDA provided helpful guidance during both model development and documentation and their suggestions were appreciatively considered in preparation of this report.

PAB-249

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF FIGURES (Cont'd)

LIST OF TABLES

INTRODUCTION

The ACDA Tactical Air Campaign Model (ATACM) is a computer model designed and built for the Arms Control and Disarmament Agency (ACDA) for use in analyzing the impacts of various Mutual Balanced Force Reduction (MBFR) proposals upon a tactical airwar in Europe between NATO and Warsaw Pact forces. The design of ATACM was based upon the findings of a survey (Reference 1) of existing tactical air models conducted by KETRON to assess the applicability of existing models to ACDA's requirements. Results of the survey indicated the need for a new model incorporating the most desirable features of existing models (e.g., TAC CONTENDER, VECTOR, OPTSA, DYGAM) into a rigorous optimization framework allowing more aircraft types and a wider selection of aircraft missions.

As a realization of the survey's recommendations, ATACM models an air campaign as a zero-sum staged game between opposing forces and employs dynamic programming to solve this game for approximate, optimal aircraft allocation strategies for both sides at each stage of the campaign. Because of the economies associated with the optimization methodology used, ATACM offers many features previously not practical in other optimizing models. Specifically, ATACM permits:

- as many as four user-defined aircraft types per side and as many as eight different missions per aircraft type

- automatic generation of approximate, optimal, enforceable aircraft allocation strategies as a function of stage for any subset of the missions for which user-specified fractions are not supplied. The user may specify all, part, or none of the allocation fractions for a given aircraft type and the model generates optimal values for those fractions not specified.

- calculation of firm upper and lower bounds on the objective function value associated with the enforceable strategies employed

- option to use a weighted sum of three different objective functions as the criterion for generating the optimal strategies

- option to individually weight the Blue and Red contributions to these objective functions as a function of stage

- option to specify fractional or numerical reinforcements for any aircraft type as a function of stage

Following sections present a description of how the problem of tactical air warfare is formulated in ATACM, a description of the attrition relationships used to evaluate the outcomes of air-to-air, air-to-ground, and ground-to-air engagements, and an outline of the optimization methodology used to generate optimal enforceable strategies and objective function bounds. Appendix A is a user's guide for ATACM which describes the model's inputs, outputs, and general operation. Appendix B presents programming documentation and FORTRAN listings coded for the CDC6600.

PROBLEM FORMULATION

ATACM formulates a tactical air campaign as a staged game between opposing Blue and Red air and ground forces. It generates for each side, and for each stage of the war, strategies which optimize the utilization of these forces over the length of the campaign. Figure 1 presents a graphical representation of a general staged game which depicts the roles of those essential elements which will be described in detail in following sections.

In Figure 1, for each stage or time period of the campaign, vertical planes represent the state space of possible beginning and ending force levels for the opposing sides. Corresponding to each point in the beginning state space for a given stage, a game matrix can be constructed with m_t and n_t strategies available to Blue and Red respectively. The numbers of strategies, m_t and n_t, are a function of stage, while the one-stage payoffs in the game matrix depend upon the starting resource levels, the objective function chosen as a measure of overall performance, and the strategies selected by both sides.

For a given strategy selection at the beginning of stage 1, assessment relationships determine the value of the payoff and the attrition or losses suffered by both sides as a result of the one-stage battle. The dashed line in Figure 1 from stage 1 to stage 2 depicts the effect of this attrition and shows that the starting force levels for stage 2 will, barring reinforcements, generally be less than those for stage 1. The decision facing both sides at stage 2 is analogous to that at stage 1, the only possible difference being the strategies available to each side and the associated one-stage payoffs. Once again strategies are chosen, the objective function is incremented by the corresponding payoff, attrition relationships map the starting force level for the current stage into a new point in the state space for the next stage, etc. This process is repeated for the number of stages in the game, and the value of the objective function summed over all stages determines the outcome of the campaign.

Within the general framework depicted in Figure 1, the key elements in the formulation of a tactical air war as a staged game are the forces, strategies, and objective functions. The way these elements are defined in ATACM is described in the following paragraphs of this section. The key elements in the solution of a tactical air war include the assessment methodology, used to compute payoffs and attrition, and the optimization methodology used to select optimal strategies at each stage of the campaign. These elements are discussed in following sections.

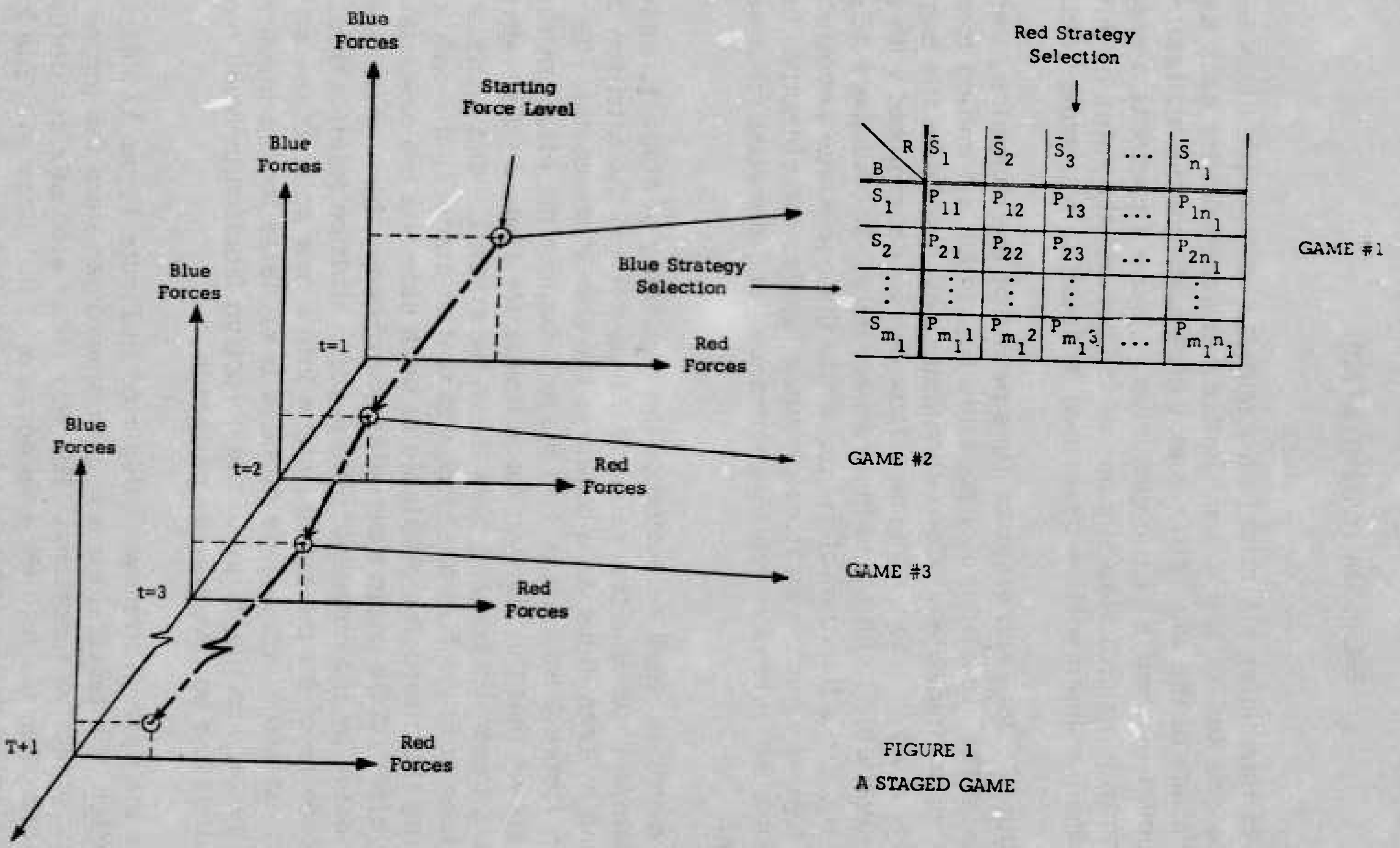

Blue Forces
Blue Forces
Blue Forces
Blue Forces
Starting Force Level
Red Strategy Selection
Blue Strategy Selection
t=1
t=2
t=3
T+1
Red Forces
Red Forces
Red Forces
Red Forces
GAME #1
GAME #2
GAME #3
R
B
$\bar{S}_1$ $\bar{S}_2$ $\bar{S}_3$ ... $\bar{S}_{n_1}$
S_1 P_{11} P_{12} P_{13} ... P_{1n_1}
S_2 P_{21} P_{22} P_{23} ... P_{2n_1}
S_{m_1} $P_{m_1 1}$ $P_{m_1 2}$ $P_{m_1 3}$ ... $P_{m_1 n_1}$

FIGURE 1
A STAGED GAME

OPPOSING FORCES AND MISSIONS

In the ATACM formulation, opposing forces consist of aircraft, SAMs, and ground divisions deployed in the stylized scenario depicted in Figure 2. A single airbase on each side serves as the staging area for all air missions flown against the opponent's SAMs, ground troops, or airbase. SAMs, which may be interpreted as any type of surface-to-air defense weapons, are segregated into forward and rear components corresponding to the location of the area they defend. Ground troops are defined in terms of homogenous divisions fighting on either side of a single-sector front (FEBA).

Aircraft

As many as four aircraft types can be assigned to either side and each of these types may be assigned as many as eight missions chosen from those listed in Table 1. In the course of a battle, forward and rear SAM suppressors (FSS and RSS) deploy before other aircraft and suppress the enemy's SAM sites. Afterward, aircraft assigned to fly close air support (CAS) attack the enemy's ground troops, reduce their total firepower, and directly influence the movement of the FEBA. Airbase attack (ABA) aircraft attack the opponent's airbase, destroy aircraft parked in shelters or on the open airfield, and thus reduce the enemy's effectiveness later in the war. Close air support and airbase attack escorts (CASE and ABAE) accompany the CAS and ABA aircraft on their attack missions and provide them protection from the opponent's battlefield and airbase defenders (BD and ABD). Finally, aircraft assigned to the Nothing mission remain on the ground during the battle because of the strength of opposing forces, maintenance requirements, unpreparedness due to a surprise attack, etc.

SAMs

The mission of the forward and rear SAMs is to defend the ground troops and airbase by attacking and destroying enemy aircraft. In prosecuting this mission, forward SAMs attack FSS, CAS, and CASE aircraft in direct defense of the ground troops, as well as RSS, ABA, and ABAE aircraft which must fly over the forward SAMs to reach their targets in the rear. Rear SAMs attack only those aircraft flying RSS, ABA, and ABAE missions.

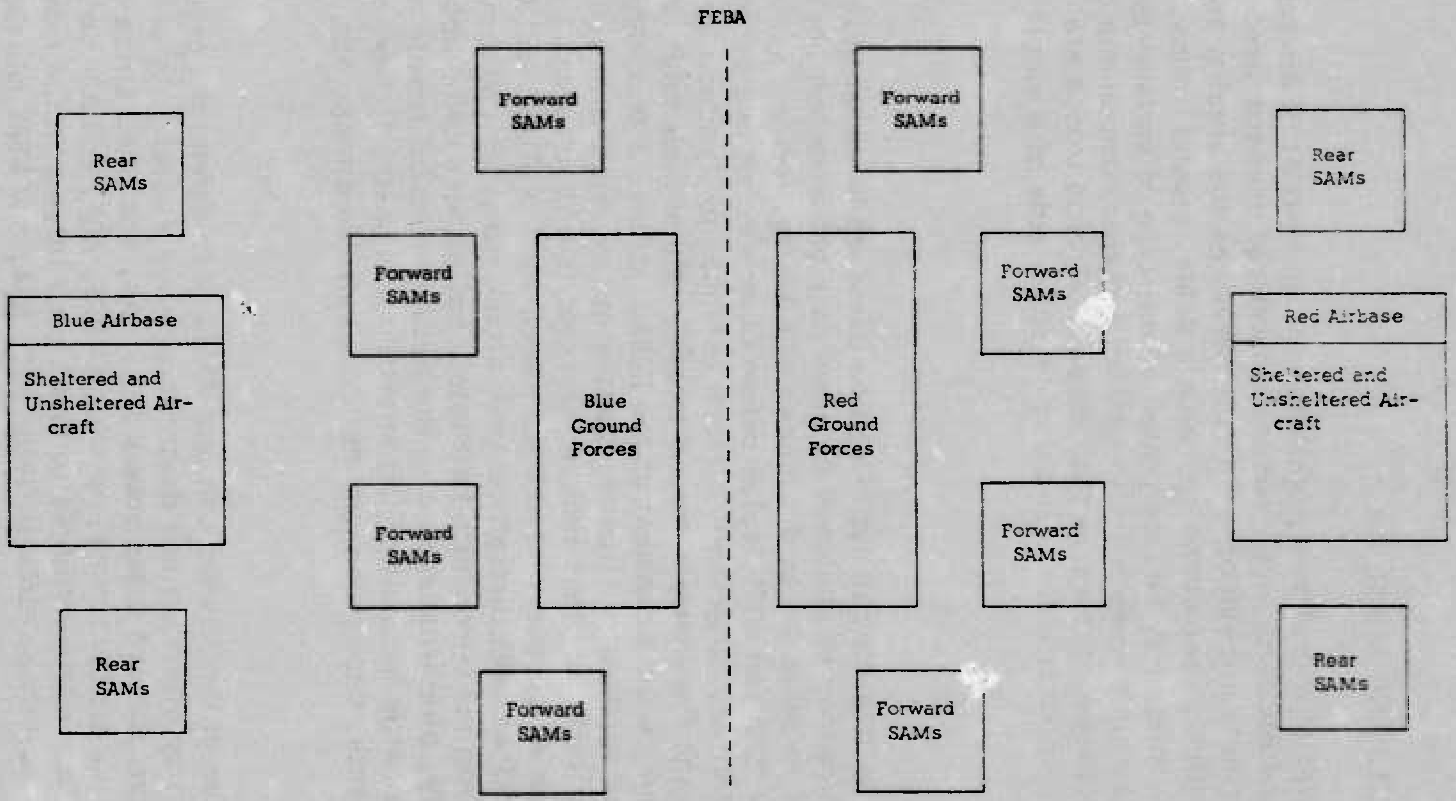


FIGURE 2
ATACM SCENARIO

PAB-249

TABLE 1

AIRCRAFT MISSIONS PERMITTED IN ATACM

Acronym	Mission
CAS	Close Air Support
ABA	Airbase Attack
BD	Battlefield Defense
ABD	Airbase Defense
CASE	Close Air Support Escort
ABAE	Airbase Attack Escort
FSS	Forward SAM Suppression
RSS	Rear SAM Suppression
Nothing	No assigned mission

Ground Troops

Because ATACM was designed as a tool for studying the effects of different numbers and types of aircraft upon the outcome of an air campaign, the ground representation is simplistic and serves primarily as an input to the figure of merit used in the optimization process. Ground troops are segregated into homogeneous divisions with each division assigned a maximum firepower score. Successful CAS sorties flown against the enemy troops reduce this maximum firepower by a specified amount, and FEBA movement is then calculated as a user-specified function of the ratio of the total net firepowers delivered by Blue and Red respectively.

STRATEGIES

Given the forces and missions described above, the strategy a commander uses during a given time period or stage is a fractional allocation of all aircraft to missions. For example, if only one aircraft type is available to the Blue commander, and this aircraft can prosecute four missions selected from Table 1, the set of possible strategies from which he may choose can be characterized as the set of all 4-tuples whose elements are positive fractions which sum to one. Examples include (.5, 0, .5, 0), (.5, 0, .2, .3), (.25, .25, .25, .25), etc. In the general case of s missions, s-tuples representing possible fractional allocations for one aircraft type are called decision vectors.

If the Blue commander has only one aircraft type, the sets of possible decision vectors and strategies are identical. If two aircraft types are available to the Blue side, the set of possible strategies corresponds to the set of all possible decision vector pairs with the first decision vector representing allocations for aircraft type 1, the second allocations for aircraft type 2. An example of a strategy for two aircraft types, each with four possible missions, would be

$$\big((.5,\ 0,\ .5,\ 0),\ (.5,\ .2,\ .2,\ .1)\big)$$

Analogously, possible strategies for a side with three aircraft types can be represented as decision vector triples, etc.

To limit the number of decision vectors (and thus strategies) from which ATACM must choose an optimal allocation, a parameter called a minimum allocation fraction is specified for each aircraft type. The minimum allocation fraction for an aircraft type is the smallest fractional unit which can be assigned to any mission. In the case of an aircraft type with four assigned missions, a minimum allocation fraction of .5 limits the set of possible decision vectors to the following ten.

(1, 0, 0, 0)	(0, .5, 0, .5)
(.5, 0, 0, .5)	(0, .5, .5, 0)
(.5, 0, .5, 0)	(0, 0, 1, 0)
(.5, .5, 0, 0)	(0, 0, .5, .5)
(0, 1, 0, 0)	(0, 0, 0, 1)

Correspondingly, the number of strategies available to a side with two, three, or four such aircraft types would be 100, 1000, or 10,000 respectively. In general, the number of possible decision vectors V for an aircraft type with s missions and a minimum allocation fraction equal to 1/t is given by

$$V = \frac{(s + t - 1)!}{(s - 1)!\, t!} \tag{1}$$

In addition to aircraft types, missions assigned, and minimum allocation fractions, ATACM permits one other important specification which determines the set of strategies available for a particular stage of the conflict. The user is allowed to specify, for any stage, a fixed assignment of aircraft to mission in terms of a multiple of the minimum allocation fraction. Looking at the previous example, the user can force half of the aircraft to prosecute the first mission assigned by specifying the first element in the corresponding decision vector to always equal .5. In that case, the set of possible decision vectors for the specified stage and aircraft type reduces to the following subset of those shown above:

(.5, 0, 0, .5)
(.5, 0, .5, 0)
(.5, .5, 0, 0)

The set of all strategies generated from these three decision vectors will reflect the specified allocation, and the strategy selected from this set by ATACM will optimize remaining allocations over those missions for which fractions are not specified.

As can be seen from this example, as more and more fractions are specified, the number of possible decision vectors and strategies decreases, and the process of strategy selection optimizes over fewer and fewer missions. In the extreme case, where all fractions in all decision vectors are specified, the set of strategies available at each stage reduces to a single strategy. Strategy selection then becomes a vacuous operation, and the net effect is an evaluation of an air campaign using a user-specified strategy at each stage. Thus, depending upon the number of mission allocations specified, ATACM can be used as an optimization, sub-optimization, or strategy-specified model.

OBJECTIVE FUNCTIONS

ATACM permits the user to select any linear combination of three objective functions to be used as the overall measure of the opposing forces' performance during an air campaign. Specifically, the overall objective function used as the criterion for strategy selection can be expressed as

$$F = w_1 f_1 + w_2 f_2 + w_3 f_3 \qquad (2)$$

where f_1 = difference of total Blue minus total Red CAS firepower

f_2 = difference of total Blue minus total Red (CAS firepower + ground firepower)

f_3 = total FEBA movement computed as a user-specified function of the ratio of Blue's total ground firepower to Red's

and w_j = user specified weight on f_j, $j = 1, 2,$ or 3.

By appropriate choice of the w_j, the user can optimize using any one of the f_j, or he can generate hedging strategies -- those not precisely optimal for any single criterion but instead optimal for several criteria at once -- by specifying F to be a combination of the f_j. Regardless of what F is specified, f_1, f_2, and f_3 are also computed and recorded individuaaly making it possible to simultaneously monitor the effects of different optimization criteria on each of the objective functions.

In addition to the weights w_1, w_2, and w_3, ATACM also permits the user to weight the Blue and Red components of f_1, f_2, and f_3 by stage. To illustrate, each f_j can be expanded as follows:

$$f_1 = \sum_{t=1}^{T+1} b_t CAS_{Bt} - r_t CAS_{Rt} \qquad (3)$$

$$f_2 = \sum_{t=1}^{T+1} b_t TFP_{Bt} - r_t TFP_{Rt} \qquad (4)$$

$$f_3 = \sum_{t=1}^{T+1} \frac{b_t + r_t}{2} \text{ FEBA}_t \tag{5}$$

where T = number of stages in the campaign

$$\text{CAS}_{kt} = \begin{cases} \text{CAS firepower delivered by side } k \text{ during stage } t & \text{for } t \leq T \\ \text{residual value of undamaged aircraft on side } k \text{ at end of war} & \text{for } t+T+1 \end{cases}$$

$$\text{TFP}_{kt} = \begin{cases} \text{total firepower (ground + CAS) delivered by side } k \text{ during stage } t & \text{for } t \leq T \\ \text{residual value of undamaged aircraft on side } k \text{ at end of war} & \text{for } t+T+1 \end{cases}$$

$$\text{FEBA}_t = \begin{cases} \text{FEBA movement during stage } t & \text{for } t \leq T \\ 0 & \text{for } t=T+1 \end{cases}$$

b_t = weight on Blue's contribution to the objective function during stage t ($b_{T+1}=1$)

r_t = weight on Red's contribution to the objective function during stage t ($r_{T+1}=1$)

Depending upon the scenario being simulated, the weights b_t and r_t can be used to reflect the various effects of logistics, force readiness, pilot skills, ground terrain, etc., all as a function of stage. For example, in the case of a surprise attack by Red, the amount of firepower Blue can deliver per sortie during early stages of the war might be severely limited by logistics, readiness, etc. To simulate this situation, weights b_t specified for the first few stages of the war would be smaller than those specified for later stages.

ASSESSMENT METHODOLOGY

Returning to the general description of a staged game depicted in Figure 1, the assessment methodology computes the values of the payoffs in each game matrix and the attrition suffered by both sides as a result of each possible one-stage battle. The important considerations in describing this methodology include the types of engagements which may occur between the opposing sides, the sequence in which these engagements occur within each stage, and the relationships used to compute attrition for each type of engagement. An important finding of the survey of existing models was the general lack of agreement concerning the best way to treat each of these facets of the assessment procedure. The assessment methodology described below is a mix of those used in OPTSA and VECTOR (References 2 and 3) and consequently suffers from some of the same limitations cited in Reference 1. In consideration of these limitations, ATACM is purposely structured so that other, alternative assessment methodologies can be implemented with minimal programming effort.

TYPES OF ENGAGEMENTS

In the current version of ATACM the types of engagements permitted can be classified as air-to-air, air-to-ground, and ground-to-air. Air-to-air engagements occur when CAS and CASE aircraft engage battlefield defenders or when ABA and ABAE aircraft engage airbase defenders. Air-to-ground engagements occur when ABA aircraft attack the opponent's airbase, when SAM suppressors attack the opponent's SAMs, or when CAS aircraft deliver ordnance against the opponent's ground troops. Ground-to-air engagements occur between SAMs and opposing aircraft flying SAM suppression, CAS, CASE, ABA, or ABAE missions.

ENGAGEMENT CYCLES

The first event in each stage of the air campaign is the addition of any numerical or fractional aircraft reinforcements specified by the user. Thereafter, attrition and payoffs for each strategy pair are computed and accumulated over a specified number of equal length time periods (e.g. days) called engagement cycles. The first event in each engagement cycle is the assignment of aircraft to missions according to the strategy pair being examined. Then, using specified sortie rates per cycle, these assignments are converted into sorties which progress en masse through the engagements described below.

FSS and RSS Engagements

FSS and RSS missions depicted in Figure 3 are flown by each side before all other missions in an attempt to clear corridors for subsequent battlefield and airbase attackers. SAMs successfully attacked by suppressors are killed for the duration of the stage in which they are suppressed, but are replaced or restored at the beginning of the next stage. SAM suppressors successfully engaged by SAMs or destroyed by the opponent's airbase attackers are lost for the duration of the campaign.

CAS, CASE, and BD Engagements

The CAS, CASE, and opposing BD missions depicted in Figure 4 are flown after the SAM suppressors. All air-to-air engagements between CAS, CASE, and the opponent's BD sorties are one-on-one encounters. Excess attackers or defenders not engaged in one phase of the engagement cycle proceed unmolested to the next phase. Sorties which are engaged abort their original mission, fight the opposing aircraft, and, if not killed, return to their own airbase. Each successful CAS sortie delivers ordnance on the opponent's ground troops and reduces their total firepower (computed at the beginning of each stage as number of divisions times firepower per division) by a user-specified amount. Thus ground troops, like SAMs, can be thought of as being killed for the duration of the stage in which they are attacked, but replaced at the beginning of the next stage. CAS, CASE, or BD killed in air-to-air engagements or destroyed by the opponent's airbase attackers are lost for the duration of the campaign.

ABA, ABAE, and ABD Engagements

The ABA, ABAE, and opposing ABD missions depicted in Figure 5 are the last missions flown in each engagement cycle. All air-to-air engagements between ABA, ABAE, and the opponent's ABD sorties are one-on-one encounters treated in the same way as those described for CAS, CASE, and BD missions. Each successful ABA sortie delivers ordnance on the opponent's airbase destroying those sheltered and unsheltered aircraft specified as being vulnerable to airbase attack. Shelters are assigned to vulnerable aircraft types in proportion to their relative numbers. Shelters are not destroyed; damaged shelters are

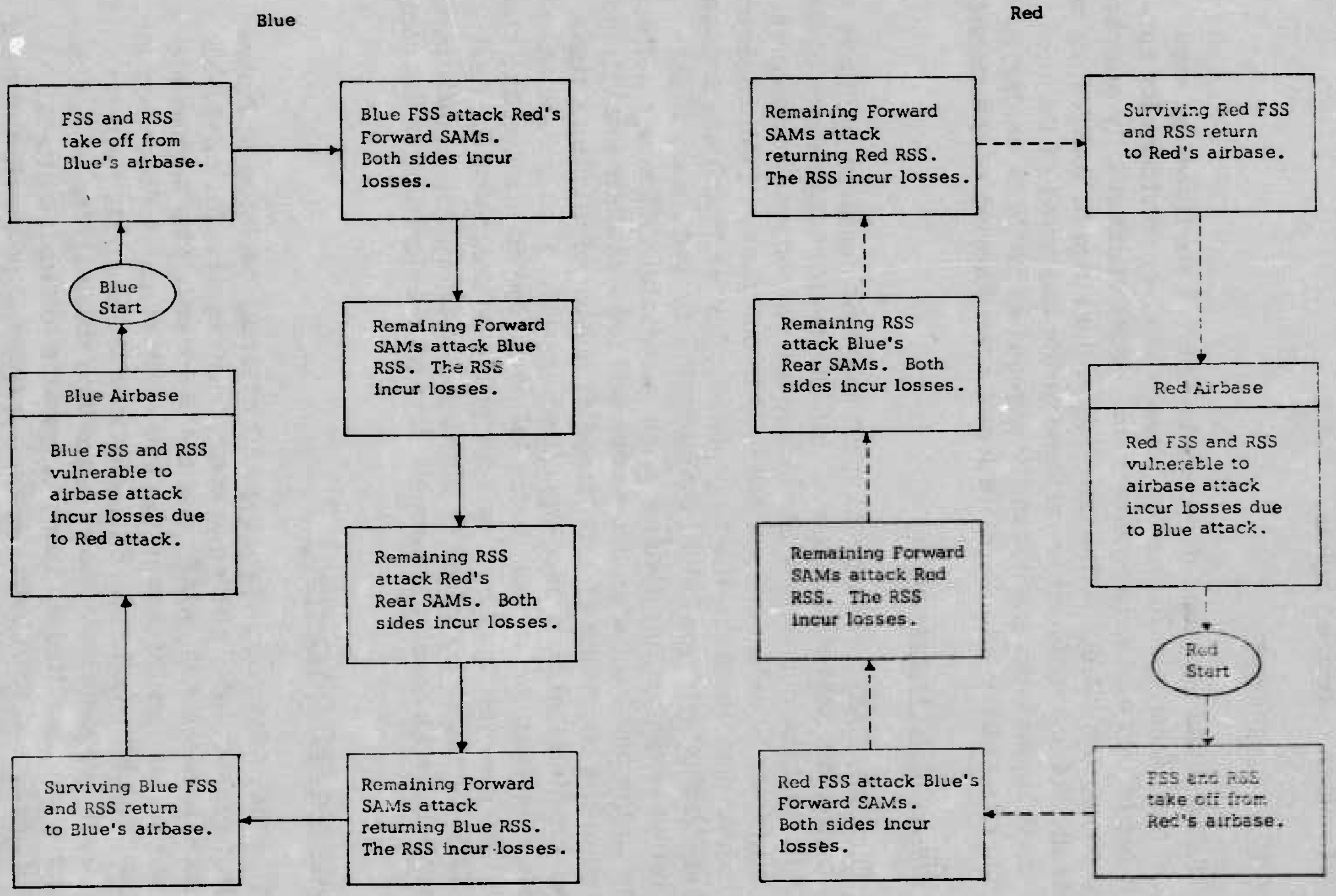

Blue
Red
FSS and RSS take off from Blue's airbase.
Blue FSS attack Red's Forward SAMs. Both sides incur losses.
Remaining Forward SAMs attack returning Red RSS. The RSS incur losses.
Surviving Red FSS and RSS return to Red's airbase.
Blue Start
Remaining Forward SAMs attack Blue RSS. The RSS incur losses.
Remaining RSS attack Blue's Rear SAMs. Both sides incur losses.
Blue Airbase
Blue FSS and RSS vulnerable to airbase attack incur losses due to Red attack.
Red Airbase
Red FSS and RSS vulnerable to airbase attack incur losses due to Blue attack.
Remaining RSS attack Red's Rear SAMs. Both sides incur losses.
Remaining Forward SAMs attack Red RSS. The RSS incur losses.
Red Start
Surviving Blue FSS and RSS return to Blue's airbase.
Remaining Forward SAMs attack returning Blue RSS. The RSS incur losses.
Red FSS attack Blue's Forward SAMs. Both sides incur losses.
FSS and RSS take off from Red's airbase.

FIGURE 3

FSS AND RSS ENGAGEMENTS DURING AN ENGAGEMENT CYCLE

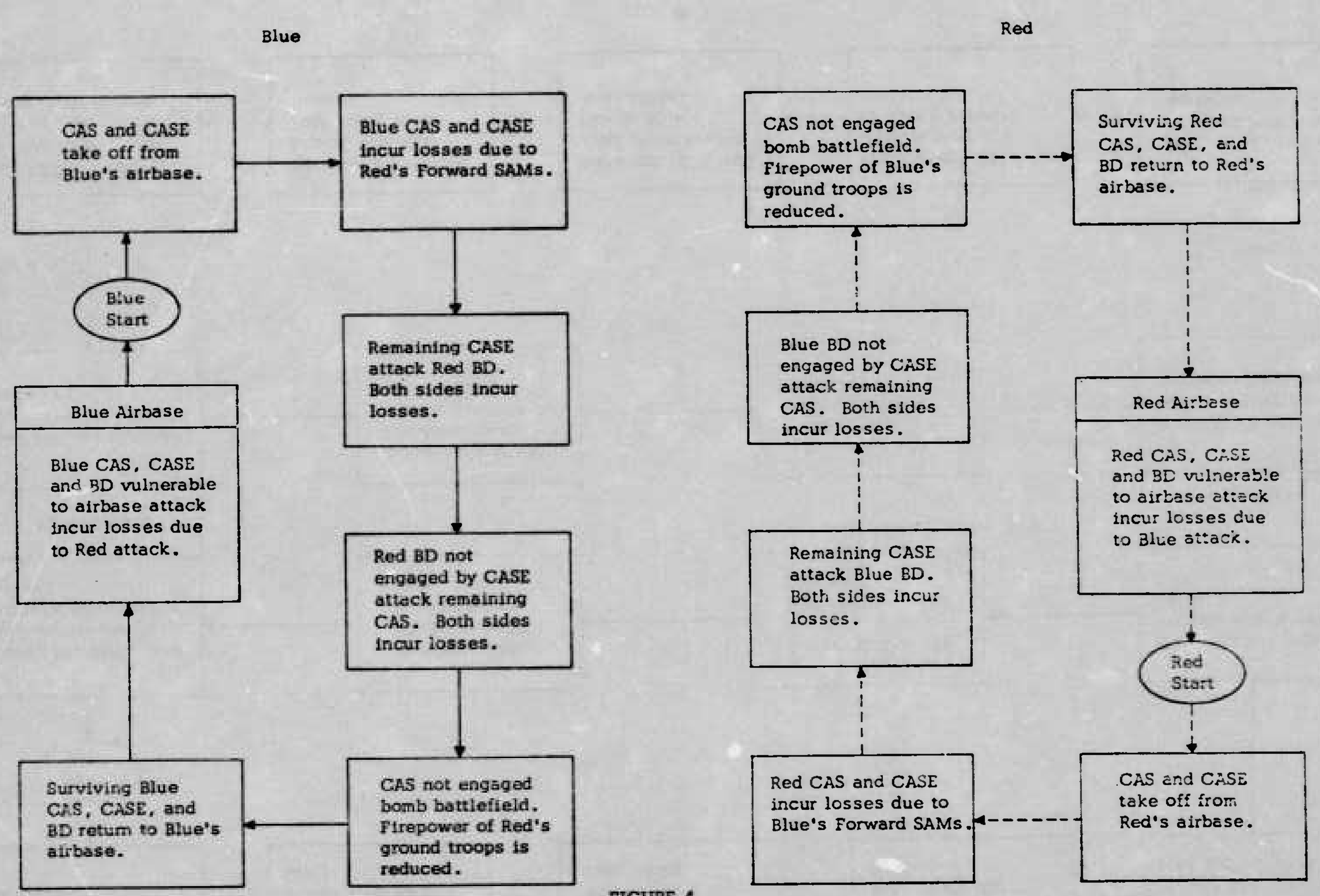

Blue
Red
CAS and CASE take off from Blue's airbase.
Blue CAS and CASE incur losses due to Red's Forward SAMs.
Remaining CASE attack Red BD. Both sides incur losses.
Red BD not engaged by CASE attack remaining CAS. Both sides incur losses.
CAS not engaged bomb battlefield. Firepower of Red's ground troops is reduced.
Surviving Blue CAS, CASE, and BD return to Blue's airbase.
Blue Airbase
Blue CAS, CASE and BD vulnerable to airbase attack incur losses due to Red attack.
Blue Start
CAS and CASE take off from Red's airbase.
Red CAS and CASE incur losses due to Blue's Forward SAMs.
Remaining CASE attack Blue BD. Both sides incur losses.
Blue BD not engaged by CASE attack remaining CAS. Both sides incur losses.
CAS not engaged bomb battlefield. Firepower of Blue's ground troops is reduced.
Surviving Red CAS, CASE, and BD return to Red's airbase.
Red Airbase
Red CAS, CASE and BD vulnerable to airbase attack incur losses due to Blue attack.
Red Start

FIGURE 4

CAS, CASE, AND BD ENGAGEMENTS DURING AN ENGAGEMENT CYCLE

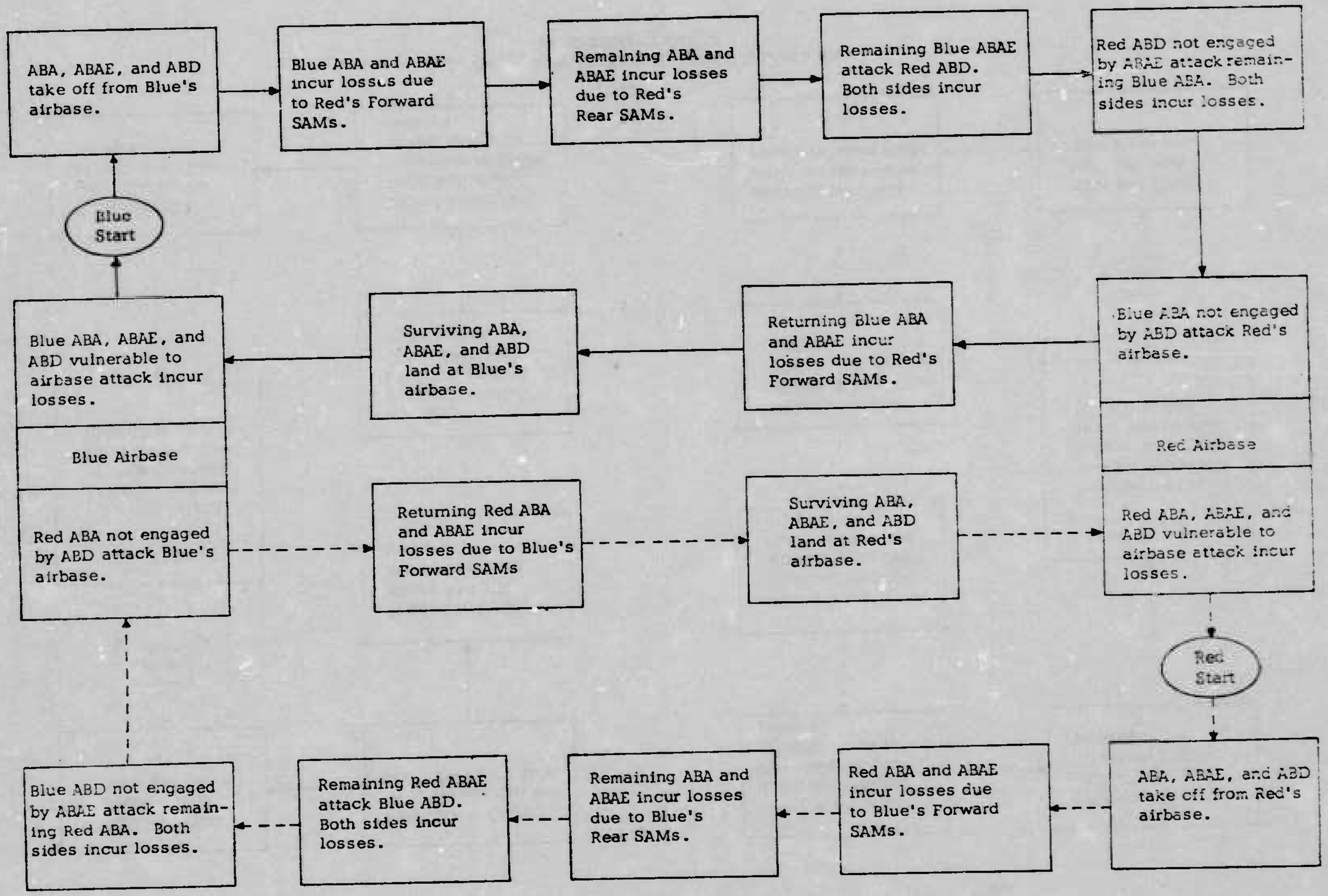

ABA, ABAE, and ABD take off from Blue's airbase.
Blue ABA and ABAE incur losses due to Red's Forward SAMs.
Remaining ABA and ABAE incur losses due to Red's Rear SAMs.
Remaining Blue ABAE attack Red ABD. Both sides incur losses.
Red ABD not engaged by ABAE attack remaining Blue ABA. Both sides incur losses.
Blue Start
Blue ABA, ABAE, and ABD vulnerable to airbase attack incur losses.
Blue Airbase
Red ABA not engaged by ABD attack Blue's airbase.
Surviving ABA, ABAE, and ABD land at Blue's airbase.
Returning Blue ABA and ABAE incur losses due to Red's Forward SAMs.
Blue ABA not engaged by ABD attack Red's airbase.
Red Airbase
Red ABA, ABAE, and ABD vulnerable to airbase attack incur losses.
Returning Red ABA and ABAE incur losses due to Blue's Forward SAMs
Surviving ABA, ABAE, and ABD land at Red's airbase.
Red Start
ABA, ABAE, and ABD take off from Red's airbase.
Red ABA and ABAE incur losses due to Blue's Forward SAMs.
Remaining ABA and ABAE incur losses due to Blue's Rear SAMs.
Remaining Red ABAE attack Blue ABD. Both sides incur losses.
Blue ABD not engaged by ABAE attack remaining Red ABA. Both sides incur losses.

FIGURE 5

ABA, ABAE, AND ABD ENGAGEMENTS DURING AN ENGAGEMENT CYCLE assumed to be repaired by the beginning of the next cycle. ABA, ABAE, or ABD killed in air-to-air engagements or destroyed by the opponent's airbase attackers are lost for the duration of the campaign.

ATTRITION RELATIONSHIPS

During the course of the engagement cycles just described, all aircraft casualties due to air-to-air and ground-to-air engagements are computed in terms of sorties lost. Surviving sorties, which successfully return to their airbase at the end of each cycle, are converted to aircraft before they are subjected to airbase attack by dividing numbers of surviving sorties by the sortie rates. The mathematical relationships used to evaluate sortie losses as well as the attrition resulting from air-to-ground engagements are described below.

Air-to-Air Engagements

In general, for each possible air-to-air engagement shown in the blocks of Figures 4 and 5, sorties on one side assigned to a generic attack mission, engage, in one-on-one encounters, sorties on the opposing side assigned to a generic defense mission. Excess attack sorties not engaged in one phase of the air-to-air war proceed to the next phase; sorties that are engaged abort their assigned mission, prosecute the air-to-air engagement and, if not killed, return immediately thereafter to their airbase. To describe how sortie attrition on both sides is computed in such an engagement, let

A_i	=	number of attack sorties in the current engagement flown by aircraft of type i
D_j	=	number of defense sorties in the current engagement flown by aircraft of type j
A_i^k	=	number of the A_i killed in the current engagement
D_j^k	=	number of the D_j killed in the current engagement
p_{ij}	=	probability an attack sortie of type i is killed by a defense sortie of type j in a one-on-one encounter
q_{ji}	=	probability a defense sortie of type j is killed by an attack sortie of type i in a one-on-one encounter

To compute A_i^k and D_j^k, the total numbers of attack and defense sorties are first used to compute E, the number of one-on-one encounters, as

$$E = \min \left(\sum_i A_i , \sum_j D_j \right) \tag{6}$$

The allocation of this total to individual aircraft types is then computed proportionally as

$$E_{ij} = E \left(\frac{A_i}{\sum_i A_i} \right) \left(\frac{D_j}{\sum_j D_j} \right) \tag{7}$$

where E_{ij} represents the number of one-on-one encounters between attackers of type i and defenders of type j.

Finally, expected numbers of killed attack and defense sorties flown by aircraft of types i and j respectively are computed as

$$A_i^k = \sum_j E_{ij} \, p_{ij} \tag{8}$$

and

$$D_j^k = \sum_i E_{ij} \, q_{ji} \tag{9}$$

Air-to-Ground Engagements

Of the three types of air-to-ground engagements considered in ATACM, only the SAM suppression and airbase attack missions directly produce losses among the opponent's forces. As described in the discussion of Figure 4, CAS sorties reduce the opponent's ground firepowei by a simple subtractive rule which indirectly reflects ground losses. By contrast, SAM and parked aircraft losses are computed using exponential relationships relating the kill probabilities and numbers of attackers (SAM suppressors or ABA) to the number of opponents (SAMs or parked aircraft).

SAM Losses

To derive the expression for SAM losses produced by generic (FSS or RSS) SAM suppression sorties, let

A_i = number of SAM suppression sorties in the current engagement flown by aircraft of type i

D = number of opposing SAMs

D^k = number of D killed in the current engagement

q_i = probability a SAM is killed by a SAM suppressor sortie of type i

To compute D^k, the numbers of SAM suppression sorties flown are used as weights to compute an average probability of kill

$$\bar{q} = \frac{\sum_i A_i q_i}{\sum_i A_i} \qquad (10)$$

This probability, along with the numbers of suppression sorties and SAMs, is used to compute D^k as

$$D^k = D \left(1 - \exp\left(- \bar{q} \frac{\sum_i A_i}{D}\right) \right) \qquad (11)$$

Losses Due to ABA

The attrition relationship for computing the effect of ABA sorties on parked aircraft is analogous to Equation (11), the only difference being the number of opponent types. Specifically, let -

A_i = number of ABA sorties in the current engagement flown by aircraft of type i

D_j = number of parked aircraft of type j vulnerable to airbase attack. Vulnerable aircraft are assigned to shelters in proportion to their relative numbers in the inventory; j=1 corresponds to sheltered aircraft, j=2 to unsheltered.

D_j^k = number of D_j killed in the current engagement.

q_{ji} = probability a vulnerable aircraft of type j is killed by a sortie flown by an ABA aircraft of type i

To determine D_j^k, first E_{ij}, the number of attack sorties of each type i assumed to attack vulnerable aircraft of type j, is computed proportionally as

$$E_{ij} = A_i \frac{D_j}{\sum_j D_j} \qquad (12)$$

Using the E_{ij} as weights, an average probability of killing a parked aircraft of type j is computed as

$$\bar{q}_j = \frac{\sum_i E_{ij} q_{ji}}{\sum_i E_{ij}} \qquad (13)$$

Finally, D_j^k is computed using the standard exponential expression

$$D_j^k = D_j \left(1 - \exp\left(- \bar{q}_j \frac{\sum_i E_{ij}}{D_j} \right) \right) \qquad (14)$$

<u>Ground-to-Air Engagements</u>

In ground-to-air engagements, SAMs attack opposing sorties in one-on-one encounters analogous to one-sided air-to-air engagements. If the number of opposing sorties exceeds the number of SAMs, excessive sorties are not attacked. If the number of SAMs exceeds the number of sorties, excessive SAMs are not launched. To describe the attrition produced by a generic SAM (forward or rear) attack, let

A	=	number of SAMs in the current engagement
D_j	=	number of target sorties in the current engagement flown by aircraft of type j
D_j^k	=	number of D_j killed in the current engagement
q_j	=	probability a target sortie flown by an aircraft of type j is killed by a SAM in a one-on-one encounter

To compute D_j^k, the total number of SAMs and opposing aircraft sorties are used to compute E, the number of one-on-one encounters, as

$$E = \min (A, \sum_j D_j) \qquad (15)$$

$$E_j = E \frac{D_j}{\sum_j D_j} \tag{16}$$

where E_j represents the number of one-on-one encounters between SAMs and target sorties of type j.

Finally, the expected number of killed sorties flown by aircraft of type j is computed as

$$D_j^k = E_j q_j \tag{17}$$

OPTIMIZATION METHODOLOGY

The methodology used to select optimal strategies for both sides at each stage of the campaign reflects the recommendations of the survey presented in Reference 1. The basic approach is to select enforceable strategies which individually optimize each side's minimal performance over the length of the campaign. Following paragraphs explain this optimization criterion in detail and describe the dynamic programming methodology used to implement it.

MAXMIN/MINMAX STRATEGIES AND PAYOFFS

In the standard game matrix such as that shown in Figure 1 the possible objective functions used to compute payoffs to Blue are defined such that positive payoffs indicate Blue success. Blue's objective is to choose that strategy which will produce the largest payoff, while Red's objective is to choose that strategy which will produce the smallest payoff. The approach used in ATACM is to assume both Blue and Red act conservatively in choosing their strategies. To illustrate, for a simple case, Figure 6 presents a game matrix for starting force levels in a hypothetical one-stage campaign in which Blue has five possible strategies, Red has four.

One-Stage Game

Since Blue and Red are assumed to be conservative, each chooses that strategy which will produce the most favorable outcome under the worst of circumstances -- i.e. prior knowledge of his selection by his opponent. In the case of Blue, selection of S_1 guarantees a payoff no less than 1 regardless of which strategy Red chooses. Selection of S_2 guarantees a payoff no less than 0, S_3 no less than -1, etc. These minimal payoffs (or row minimums) are shown in Figure 6 for each possible Blue selection. Assuming Red has superior intelligence, Blue would choose that strategy, S_1, which maximizes over the set of minimal payoffs. Thus S_1 is called Blue's MAXMIN strategy, and 1, the minimal payoff associated with S_1, is called the MAXMIN payoff or objective function value. The Red strategy $\bar{S}_2$ which would have to be play against S_1 to yield the MAXMIN payoff is called Red's MAXMIN strategy.

			Red MAXMIN ↓		Red MINMAX ↓	
	Blue \ Red	$\bar{S}_1$	$\bar{S}_2$	$\bar{S}_3$	$\bar{S}_4$	Row Min
Blue MAXMIN →	S_1	2	$\boxed{1}$	4	②	1
	S_2	3	5	0	0	0
Blue MINMAX →	S_3	6	-1	4	$\boxed{3}$	-1
	S_4	3	1	2	-2	-2
	S_5	0	-3	-5	1	-5
	Col Max	6	5	4	3	

FIGURE 6

ONE-STAGE GAME MATRIX

In the case of Red's selection, choice of $\bar{S}_1$ could result in a payoff as large as 6 if Blue were to play S_3, selection of $\bar{S}_2$ could result in a payoff as large as 5, $\bar{S}_3$ a payoff as large 4, and $\bar{S}_4$ a payoff as large as 3. Assuming Blue has superior intelligence, conservative Red would choose that strategy, $\bar{S}_4$, which minimizes over the set of column maximums shown in Figure 6. Strategy $\bar{S}_4$ is called Red's MINMAX strategy, S_3 is called Blue's MINMAX strategy, and 3, the payoff associated with playing $\bar{S}_4$ against S_3 is called the MINMAX payoff.

If Blue were to play its MAXMIN strategy and Red its MINMAX strategy, the payoff which would result is 2. This MAXMIN vs. MINMAX payoff will always be greater than or equal to the MAXMIN payoff (in this case 1) and less than or equal to the MINMAX payoff (in this case 3).

From this example, it should be clear how the optimization criterion used in ATACM would be applied for a one-stage campaign. Given the starting force levels, strategies, and objective function specification, payoffs in the corresponding game matrix would be computed using the assessment methodology described in the previous section. Blue's MAXMIN strategy and payoff would be computed by maximizing over row minimums while Red's MINMAX strategy and payoff would be computed by minimizing over column maximums. Output would consist of the Blue MAXMIN strategy, the Red MINMAX strategy, the lower and upper MAXMIN/MINMAX objective function bounds, and the actual MAXMIN vs. MINMAX payoff which would result if both sides played their conservative strategies.

Multi-Stage Game

To understand how the MAXMIN/MINMAX strategy selection procedures for a one-stage game extend to a multi-stage game, it is helpful to present an alternative representation of a staged game called an extended game as shown in Figure 7. In an extended game representation of a T-stage air campaign, Blue extended strategies are T-tuples whose t^{th} element is a strategy selected from the set of one-stage strategies $S_1, S_2, \ldots, S_{m_t}$ available to Blue at the t^{th} stage of the campaign. In other words, the first element in an extended strategy for Blue is the strategy Blue would use for stage 1, the 2nd element is Blue's strategy for stage 2, ..., and the T^{th} element is Blue's strategy for the last stage of the campaign. The number of possible extended

FIGURE 7

EXTENDED GAME REPRESENTATION
OF A T-STAGE GAME

Blue \ Red	$\overline{ES}_1$	$\overline{ES}_2$	...	$\overline{ES}_N$
ES_1	TP_{11}	TP_{12}	...	TP_{1N}
ES_2	TP_{21}	TP_{22}	...	TP_{2N}
⋮	⋮	⋮		⋮
ES_M	TP_{M1}	TP_{M2}	...	TP_{MN}

ES_i = Blue Extended Strategy = $(S_{i_1}, S_{i_2}, \ldots, S_{i_T})$

where S_{i_t} = Blue strategy for stage t

$\overline{ES}_j$ = Red Extended Strategy = $(\bar{S}_{j_1}, \bar{S}_{j_2}, \ldots, \bar{S}_{j_T})$

where $\bar{S}_{j_t}$ = Red strategy for stage t

$M = m_1 \cdot m_2 \cdot \ldots \cdot m_T$ $\quad$ $N = n_1 \cdot n_2 \cdot \ldots \cdot n_T$

TP_{ij} = Total Payoff produced by playing ES_i against $\overline{ES}_j$

strategies, M, from which Blue may choose is equal to

$$M = m_1 \cdot m_2 \cdot m_3 \cdot \ldots \cdot m_T \tag{18}$$

where m_t equals the number of possible Blue strategies available for stage t. Extended strategies for Red are analogous.

Payoffs in an extended game are total payoffs over the length of the campaign which result from playing a Blue extended strategy against a Red extended strategy. Looking back at Figure 1, a total payoff in the extended game depends upon the one-stage payoffs and the attrition represented by the dashed line. Each different combination of Blue-Red extended strategies corresponds to a unique attrition path from the state space at the beginning of stage 1 to the state space at the end of stage T.

Theoretically, selection of MAXMIN/MINMAX strategies in a multi-stage campaign, represented in the context of the extended game, is exactly the same as that described for a one-stage game. Indeed, for a one-stage campaign the extended game representation reduces to a one-stage game. Unfortunately, for air campaigns with reasonable numbers of strategies and stages the extended game representation is valuable only as an abstraction because of its prohibitive size. For example, if Blue and Red each had only 20 possible strategies to choose from at each stage of a three stage campaign, the corresponding extended game would be an 8000 by 8000 matrix requiring 64 million three-stage payoff evaluations. Assuming 10^{-3} seconds of computer time was required for each one-stage assessment, evaluation of the payoffs alone would require over 50 hours. Needless to say, a method other than the straightforward solution of the extended game is required to determine MAXMIN/MINMAX strategies for the general multi-stage campaign.

DYNAMIC PROGRAMMING SOLUTION

The reason the extended game representation of modest sized air campaigns becomes untractable is that it treats the problem of strategy selection for all stages as a single step process. An alternative approach, using dynamic programming, is to decompose the problem into a series of one-stage problems, similar in many respects to the way the problem was originally posed in Figure 1. To illustrate the dynamic programming approach, its necessary to define more precisely many of the concepts first presented there. For purposes of illustration, we will assume Blue and Red each have one aircraft type -- more types simply increase the dimensionality of the state space. For this case, shown in Figure 8, let

X_t	=	a point, (B_t, R_t), in the state space at the beginning of stage t corresponding to B_t and R_t aircraft available to Blue and Red respectively
$S(X_t)$	=	Blue's one-stage MAXMIN strategy selection corresponding to X_t
$\bar{S}(X_t)$	=	Red's one-stage MINMAX strategy selection corresponding to X_t
$TP(X_t)$	=	the total MAXMIN payoff associated with optimal play by Blue from state X_t at the beginning of stage t to the end of the campaign
$\overline{TP}(X_t)$	=	the total MINMAX payoff associated with optimal play by Red from state X_t at the beginning of stage t to the end of the campaign
$P_{ij}(X_t)$	=	payoff in the one-stage game matrix corresponding to selection of the i^{th} and j^{th} strategies when in state X_t
$Z_{ij}(X_t)$	=	state X_{t+1} into which selection of the i^{th} and j^{th} strategies by Blue and Red maps the state X_t

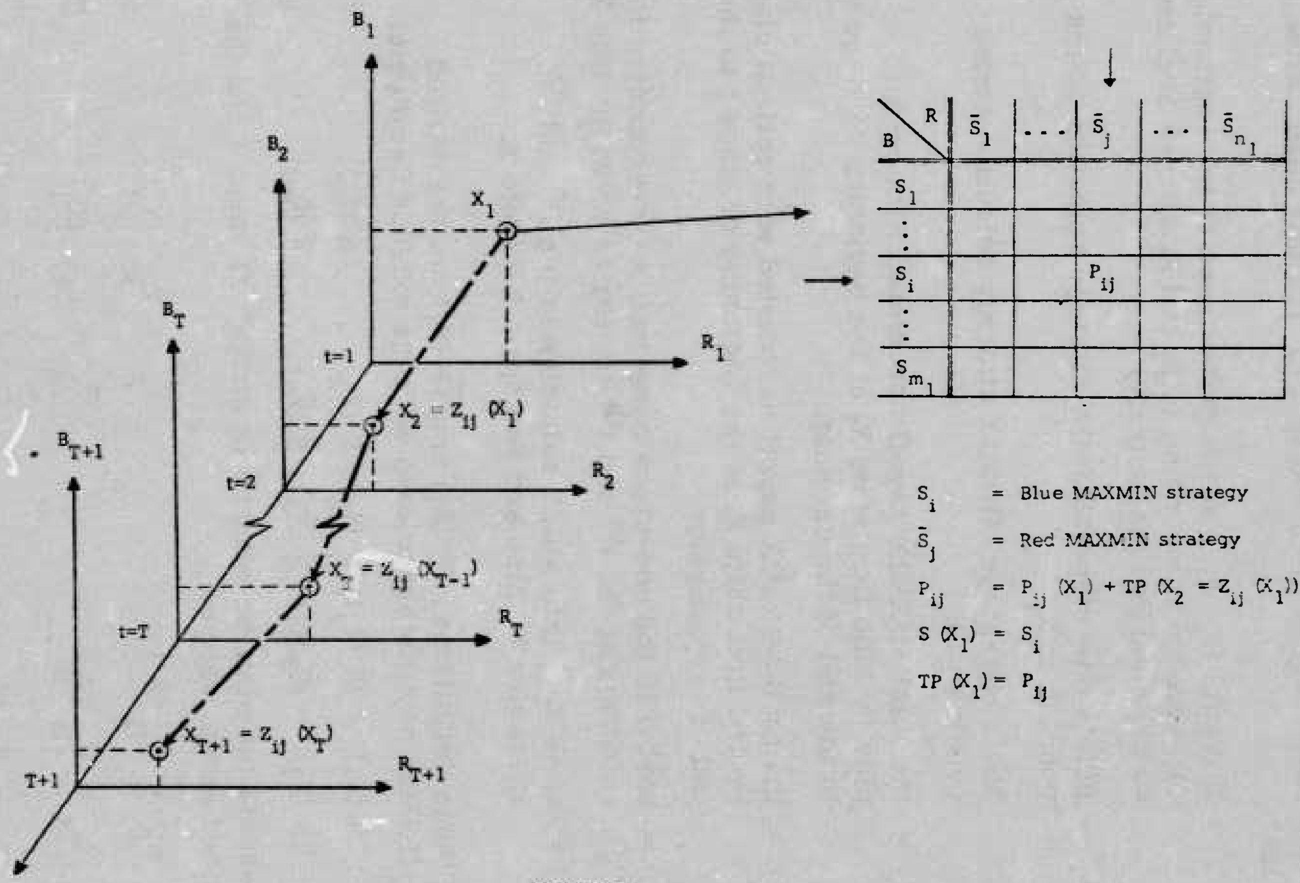
Using these definitions, $TP(X_1)$ and $\overline{TP}(X_1)$ are the desired MAXMIN and MINMAX payoffs associated with the extended strategies

$$ES(X_1) = (S(X_1), S(X_2), S(X_3), \ldots, S(X_T)) \tag{19}$$

$$\overline{ES}(X_1) = (\bar{S}(X_1), \bar{S}(\bar{X}_2), \bar{S}(\bar{X}_3), \ldots, \bar{S}(\bar{X}_T)) \tag{20}$$

where X_1 represents the starting numbers of aircraft at stage 1 and the X_t and $\bar{X}_t$ are defined recursively by

$$\begin{array}{lll} X_2 = Z_{ij}(X_1) & \bar{X}_2 = Z_{k\ell}(X_1) & \\ X_3 = Z_{ij}(X_2) & \bar{X}_3 = Z_{k\ell}(\bar{X}_2) & (21) \\ \vdots & \vdots & \\ X_T = Z_{ij}(X_{T-1}) & \bar{X}_T = Z_{k\ell}(\bar{X}_{T-1}) & \end{array}$$



S_i	=	Blue MAXMIN strategy
$\bar{S}_j$	=	Red MAXMIN strategy
P_{ij}	=	$P_{ij}\ (X_1) + TP\ (X_2 = Z_{ij}\ (X_1))$
$S\ (X_1)$	=	S_i
$TP\ (X_1)$	=	P_{ij}

FIGURE 8

DYNAMIC PROGRAMMING SOLUTION FOR MAXMIN STRATEGIES AND BOUNDS

with i = index of Blue's MAXMIN strategy for the current stage
j = index of Red's MAXMIN strategy for the current stage
k = index of Blue's MINMAX strategy for the current stage
ℓ = index of Red's MINMAX strategy for the current stage

The dynamic programming approach for solving for TP (X_1), ES (X_1), $\overline{TP}$ (X_1), and $\overline{ES}$ (X_1) separates the solution for the MAXMIN total payoff and extended strategy from the MINMAX counterparts. Although all descriptions which follow concentrate on the MAXMIN solution, the MINMAX solution technique is exactly analogous.

Idealized Approach

To compute TP (X_1) and ES (X_1) using a general dynamic programming procedure, one begins at the beginning of the last stage of the campaign and computes one-stage MAXMIN payoffs and strategies, TP (X_T) and S (X_T), for all possible states X_T. The i, j th payoff, P_{ij}, in the MAXMIN game matrix for each state X_T is given by

$$P_{ij} = P_{ij}(X_T) + TP(X_{T+1} = Z_{ij}(X_T)) \qquad (22)$$

where $P_{ij}(X_T)$ is the one-stage payoff and $TP(X_{T+1} = Z_{ij}(X_T))$ is the contribution of undamaged aircraft at the end of the war, B_{T+1} and R_{T+1}, to the value of the overall objective function (Equation 2).

After TP (X_T) and S (X_T) have been computed for all states X_T, these values are stored and the process moves backward to the beginning of stage T-1. Using the TP (X_T) just computed, the elements in the payoff matrix for each stage X_{T+1} are now given by

$$P_{ij} = P_{ij}(X_{T-1}) + TP(X_T = Z_{ij}(X_{T-1})) \qquad (23)$$

where $P_{ij}(X_{T-1})$ is the payoff contribution of the current stage and $TP(X_T = Z_{ij}(X_{T-1}))$ is the MAXMIN payoff associated with optimal play thereafter. Once again, a Blue MAXMIN strategy is computed for each game matrix along with the cumulative MAXMIN payoff TP (X_{T-1}) and each is stored for all possible states X_{T-1}. Processing moves backward to the preceding stage in time, game matrices are generated for all states, etc.

Eventually, through this iterative process, strategies and payoffs can be generated for all stages and states from stage T back to the beginning of stage 1. Since X_1, the hypothesized starting resource level, is a point in the state space at the beginning of stage 1, the solution is complete. TP (X_1) is available immediately, while the one-stage strategies which compose ES (X_1) can be retrieved from storage by tracing through the states using the attrition map Z_{ij}. Perhaps even more important, as a by-product of the dynamic programming approach, solutions for all other X_t are also available since they have been stored during the processing required to solve for X_1. By simply retrieving the stored results, solutions for all shorter campaigns less than T stages which begin with any number of aircraft on either side can be easily generated.

Problems

Precise implementation of the general dynamic programming algorithm as described above requires evaluation of one-stage game for every possible state of every possible stage. The number of such states in a reasonable sized game is huge, even for the case in which each side has only one aircraft type. For example, each side might reasonably start with 1000 aircraft, making the number of possible (B_t, R_t) pairs more than one million.

Intuitively, one would suspect states which differ by only a few aircraft would produce approximately the same solution. ATACM exploits this intuitive feel by imposing a descrete grid upon the state space at each stage in a manner analogous to that used in DYGAM (Reference 4). If a grid such as that shown in Figure 9 were imposed, a tractable approach would be to explicitly compute one-stage strategies and payoffs for only the discrete grid points using the dynamic programming procedure described above. Unfortunately, as shown in Figure 9, the Z_{ij} (X_t) for a grid point X_t would generally not lie on a grid point in the subsequent stage's state space. Since TP $(Z_{ij}$ $(X_t))$ is necessary to compute the elements in the one-stage game matrix (Equation (22)), an approximation technique is required for computing payoffs associated with points not on the grid.

Possible Approximations

One possible approximation technique for computing TP $(X_{t+1} = Z_{ij}$ $(X_t))$ would be to linearly interpolate using the explicit payoffs for grid points adjacent to X_{t+1}. It can be shown that as the grid becomes finer the strategies which would result from using linear interpolation would approach those produced by the idealized dynamic programming solution. Similarly, the payoffs produced would approach the idealized lower bound TP (X_1), but not necessarily from below. In other words, although the

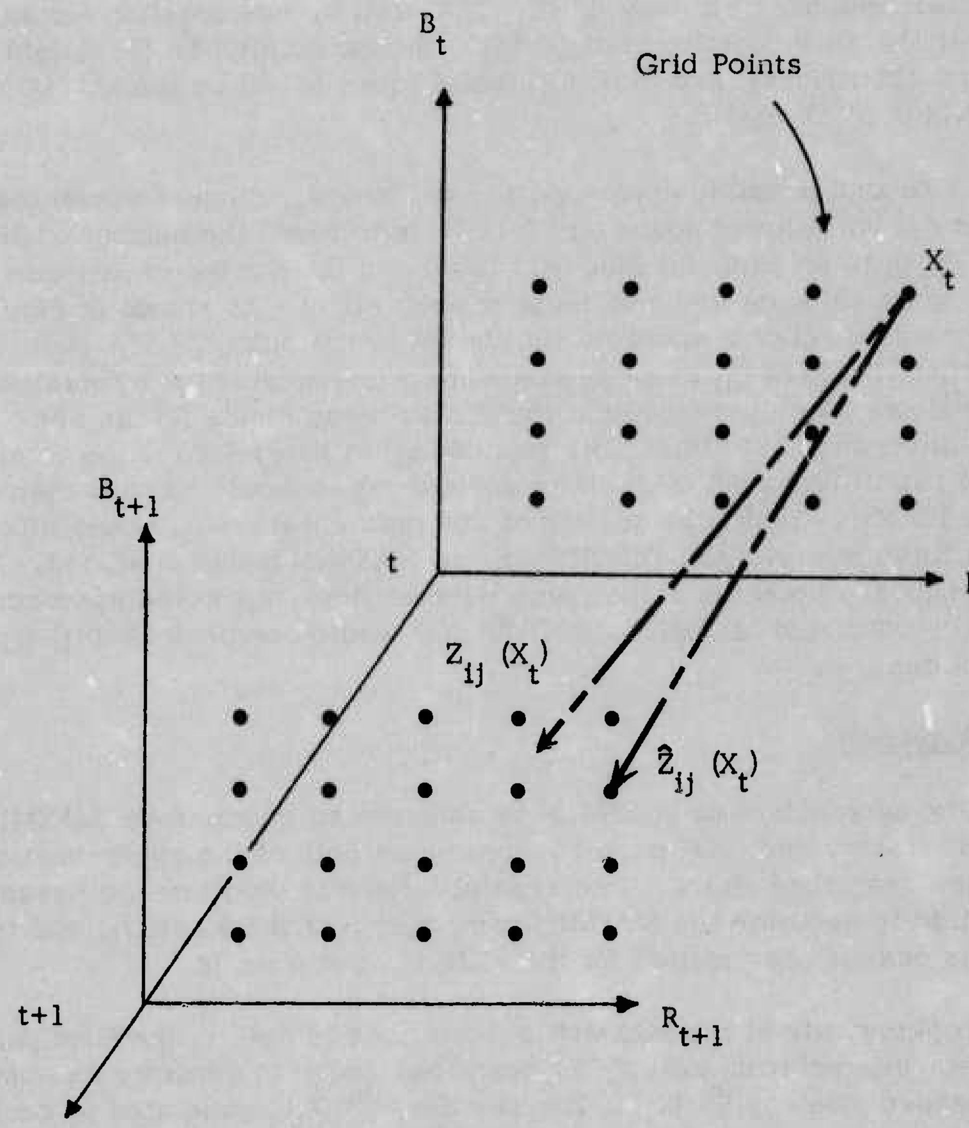


FIGURE 9

DYNAMIC PROGRAMMING
APPROXIMATIONS

approximate extended strategy $\hat{ES}$ (X_1) produced by interpolation would intuitively be a good estimate of ES (X_1), the associated $\hat{TP}$ (X_1) might be greater than TP (X_1) and thus an invalid lower bound on the MAXMIN vs. MINMAX total payoff.

A second possible approximation technique, which is guaranteed to produce a valid lower bound on TP (X_1), is to round the number of Blue aircraft down to an adjacent Blue grid level and the number of Red aircraft up to an adjacent Red grid level at each stage. As shown in Figure 9, such a rounding scheme would be equivalent to replacing TP $(Z_{ij}$ $(X_t))$ by TP $(\hat{Z}_{ij}$ $(X_t))$ where $\hat{Z}_{ij}$ is an approximate mapping which incorporates the Blue-down, Red-up rounding criteria. Because Blue's forces are rounded down and Red's forces are rounded up at every stage, the total MAXMIN payoff produced by such an approximation would be less than or equal to TP (X_1). Thus, the quality of the approximation $\hat{Z}_{ij}$ would affect only the tightness, not the validity, of the MAXMIN bound produced. As the grid imposed upon the state space became finer, $\hat{Z}_{ij}$ would approach Z_{ij} and the computed MAXMIN payoff $\hat{TP}$ (X_1) would approach TP (X_1) from below as desired.

ATACM Approach

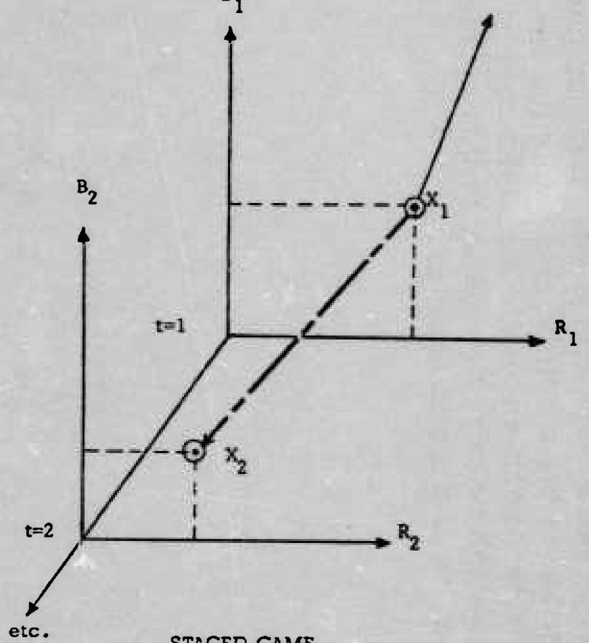
The approach used in ATACM to generate an approximate MAXMIN extended strategy and total payoff incorporates both of the approximation techniques described above. Two separate dynamic programming passes are required to generate the MAXMIN components of the solution, and two analogous passes are required for the MINMAX components.

Looking only at the MAXMIN calculations in detail, the first pass uses linear interpolation exactly as described above to generate an approximate extended strategy $\hat{ES}$ (X_1). The payoffs, $\hat{TP}$ (X_t), generated at each stage and state are used only to compute $\hat{ES}$ (X_1) and are discarded after the one-stage strategies are stored.

The second pass, designed to produce a lower bound on the MAXMIN total payoff TP (X_1), uses the $\hat{S}$ (X_t) stored during the first pass as fixed Blue strategies against which Red performs a MINMAX pass using the rounding scheme described above. As shown in Figure 10, in this second pass the game matrix for each stage and state has only one Blue strategy. Column maximums in these one-stage games are the single one-stage payoffs corresponding to each Blue strategy, and the MINMAX pass reduces to the calculation of the minimum payoff Red can achieve against $\hat{ES}$ (X_1).

PAB-249

B \ R	$\bar{S}_1$	$\bar{S}_2$	$\bar{S}_3$	...	$\bar{S}_{n_1}$
$\hat{S}(X_1)$	P_{i1}	P_{i2}	P_{i3}	...	P_{in_1}



STAGED GAME

OR

B \ R	$\overline{ES}_1$	$\overline{ES}_2$	$\overline{ES}_3$	...	$\overline{ES}_N$
$\hat{FS}(X_1)$	TP_{i1}	TP_{i2}	TP_{i3}	...	TP_{iN}

EXTENDED GAME

FIGURE 10

ALTERNATIVE REPRESENTATIONS
OF THE SECOND DYNAMIC PROGRAMMING
PASS FOR COMPUTING $\hat{TP}(X_1)$

To see that this second MINMAX pass produces a lower bound on TP (X_1), consider two cases.

- If $\hat{ES}$ (X_1) is the same as ES (X_1), and no Blue-down/ Red-up rounding is required, the best Red can do in the MINMAX pass is to generate Red's MAXMIN extended strategy, which, by definition, produces a total payoff exactly equal to TP (X_1). If rounding is required, the total payoff would clearly be less than or equal to TP (X_1).

- If $\hat{ES}$ (X_1) is not equal to ES (X_1), and rounding is not required, the minimum total payoff Red can achieve in the MINMAX pass is the row minimum corresponding to $\hat{ES}$ (X_1) in the extended game of Figure 10. Since TP (X_1) is defined as the maximum over all row minimums, TP (X_1) must be bounded from below by the MINMAX payoff. As before, if rounding is required the value of the MINMAX payoff can only be reduced.

Finally, after two analogous dynamic programming passes for computing estimates of Red's MINMAX extended strategy $\overline{ES}$ (X_1) and the upper objective function bound TP (X_1), the solution is virtually complete. All that remains is the estimation of the MAXMIN vs. MINMAX payoff for starting state X_1 which is accomplished by actually playing the one-stage strategies stored for Blue and Red at each grid point. If during this calculation a Z_{ij} (X_t) does not fall on one of the specified grid points the one-stage strategies stored for the nearest point are used. As in the case of computing the MAXMIN and MINMAX bounds, the quality of this approximation depends upon the coarseness of the grid imposed.

REFERENCES

1. R. J. Galiano and F. A. Miercort, "Results of a Survey of Tactical Air Campaign Models," Contract No. ACDA/PAB-249, KETRON, Inc., November 1974.

2. Jerome Bracken, "Two Optimal Sortie Allocation Models," Paper P-992, Institute for Defense Analyses, December 1973.

3. S. Bonder, "VECTOR-O, The Battle Model Prototype," WSEG Report 222, Weapons System Evaluation Group, December 1973.

4. Control Analysis Corporation, "Development of an Algorithm to Solve Multi-Stage Games," 24 May 1973.

APPENDIX A

ATACM USER'S GUIDE

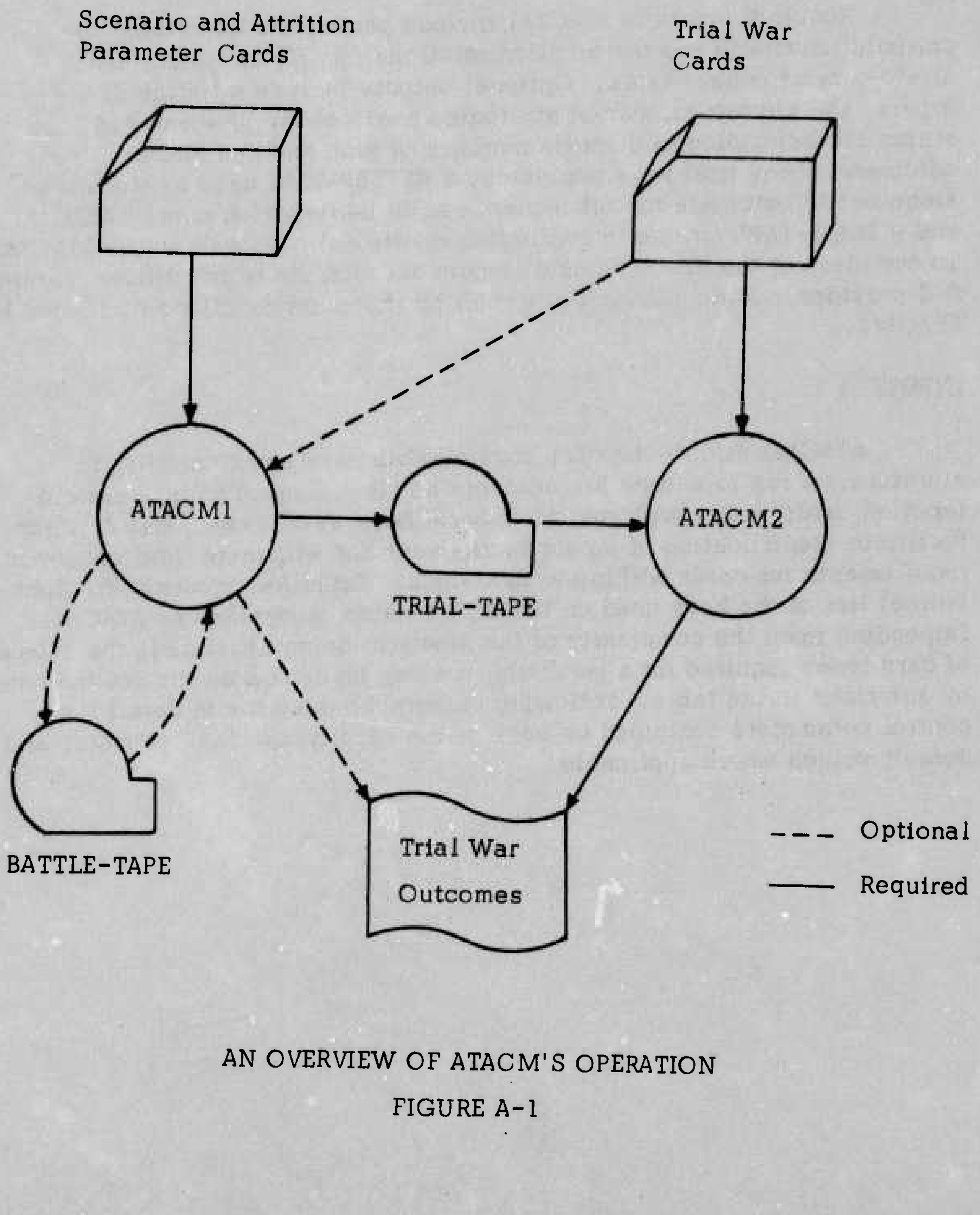
This appendix is a guide to the use of ATACM. Following sections describe the inputs, the operation, and the outputs of the model.

GENERAL OVERVIEW

The computer model ATACM was developed by the Arms Control and Disarmament Agency as a tool for studying the impact of various force postures upon the outcome of a tactical airwar in Europe between NATO and Warsaw Pact forces. ATACM treats an air campaign as a zero-sum staged gamed between opposing forces, Blue and Red, each consisting of air and ground forces. Most of the emphasis in the current version of ATACM is upon air force interactions although a simplistic representation of the ground war is also included. The model uses dynamic programming to generate for each stage and state of the campaign optimal aircraft allocation strategies and associated MAXMIN/MINMAX bounds on a user-specified objective function.

As currently implemented, ATACM consists of two FORTRAN programs, ATACM1 and ATACM2, designed for use on a CDC 6600 computer. ATACM1 generates optimal strategies and associated objective function bounds for each stage and state and uses these values to evaluate the outcomes of trial wars of specified length which are initiated with user-specified numbers of Blue and Red aircraft. In addition to the outcomes of the trial wars, ATACM1 can output to magnetic tape one-stage battle assessments (BATTLE-TAPE) for use in subsequent runs of ATACM1, and optimal strategies and bounds (TRIAL-TAPE) for use in subsequent trial war evaluations using ATACM2. To evaluate additional trial wars, ATACM2 reads the TRIAL-TAPE and uses the information it contains to process trial war requests and produce outputs identical to those which would have been produced if the requests had been evaluated during the execution of ATACM1. Thus, for a given set of scenario and attrition parameters, only a single execution of ATACM1 is necessary regardless of when or how many trial war evaluations are eventually required.

Figure A-1 presents a schematic representation of how ATACM1, ATACM2, the BATTLE-TAPE, and the TRIAL-TAPE interrelate.



AN OVERVIEW OF ATACM'S OPERATION

FIGURE A-1

ATACM1

Required inputs to ATACM1 include parameters describing the campaign scenario and the attrition relationships for air-to-air and air-to-ground engagements. Optional outputs include a listing of the inputs, the aircraft allocation strategies available to Blue and Red, the states corresponding to discrete numbers of Blue and Red aircraft, the outcomes of any trial wars requested, a BATTLE-TAPE used to store one-stage battle outcomes for subsequent use in perturbation runs of ATACM1, and a TRIAL-TAPE for use in evaluating additional trial wars using ATACM2. To complement the discussions of inputs and outputs which follow, Figure A-2 provides a logical flowchart depicting the major functions performed by ATACM1.

INPUTS

ATACM1 affords the user considerable ease and flexibility in sturcturing a run to satisfy his analysis needs. Control parameters are input on cards with identifying alphabetic keys in columns 1 thru 5 which facilitate identification of inputs by the user and eliminate rigid sequence requirements for cards within the data deck. Table A-1 presents an alphabetical list of the keys used on the inputs cards recognized by ATACM1. Depending upon the complexity of the scenario being simulated, the number of card types required for a particular run can be as few as the six indicated by asterisks in the table. Following paragraphs describe in detail the control parameters contained on each of the card types, their formats, and default values where applicable.

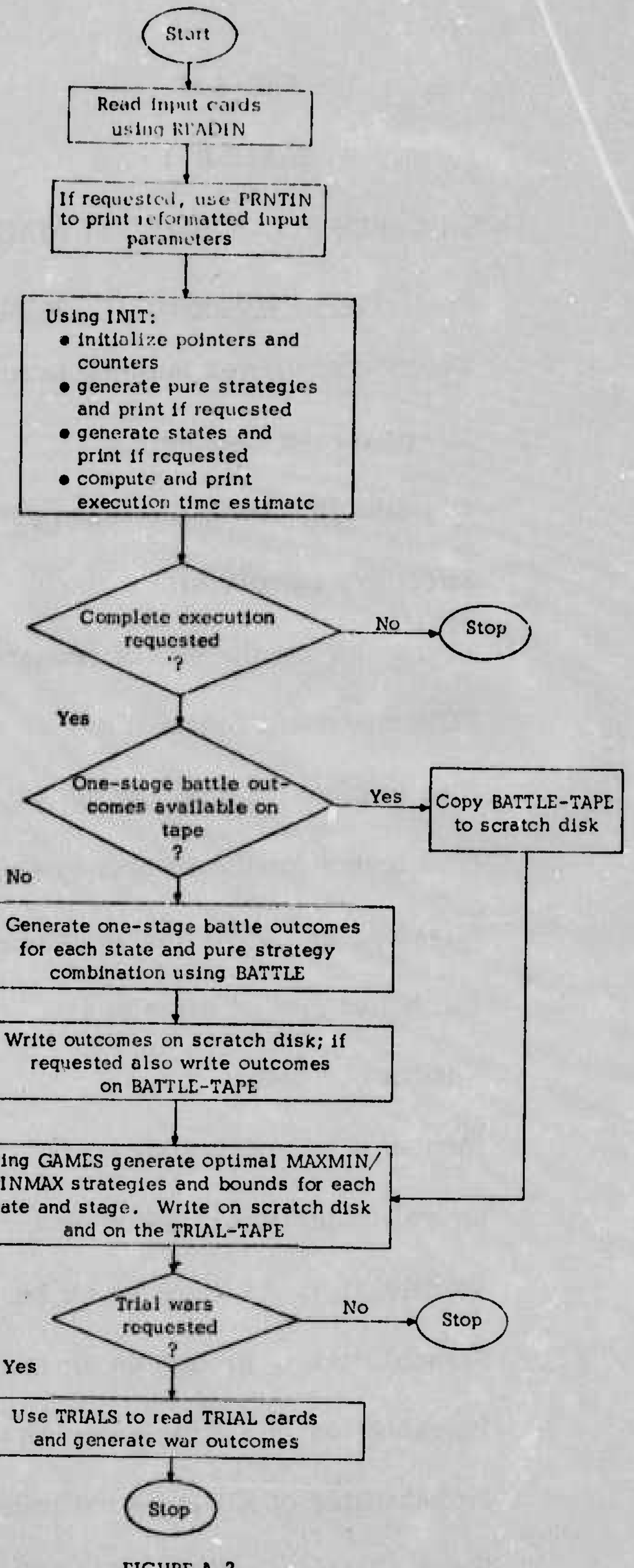

Start
Read input cards using READIN
If requested, use PRNTIN to print reformatted input parameters
Using INIT:
• initialize pointers and counters
• generate pure strategies and print if requested
• generate states and print if requested
• compute and print execution time estimate
Complete execution requested ?
No
Stop
Yes
One-stage battle outcomes available on tape ?
Yes
Copy BATTLE-TAPE to scratch disk
No
Generate one-stage battle outcomes for each state and pure strategy combination using BATTLE
Write outcomes on scratch disk; if requested also write outcomes on BATTLE-TAPE
Using GAMES generate optimal MAXMIN/MINMAX strategies and bounds for each state and stage. Write on scratch disk and on the TRIAL-TAPE
Trial wars requested ?
No
Stop
Yes
Use TRIALS to read TRIAL cards and generate war outcomes
Stop

FIGURE A-2
LOGICAL FLOWCHART OF ATACM1

TABLE A-1

INPUT CARDS RECOGNIZED BY ATACM1

Card Key	Input Parameters It Contains
ABAF	Fraction of planes vulnerable to ABA
CASF	Firepower per CAS sortie
DFRC	Division firepower reduction per CAS sortie
DIVF	Firepower per division
END*	Flag to signal the end of scenario and attrition inputs
FEBAM	FEBA movement function
FINIS	Flag to signal the end of the TRIAL cards
GRID**	Grid levels for the number of available planes
MISS**	Missions assigned and associated sortie rates
NDIV	Number of ground divisions
NSAM	Number of SAMs
NSHL	Number of air base shelters
OWGHT	Overall objective function weights
PKAA	Probabilities of killing an air base attacker
PKAD	Probabilities of killing an air base defender
PKAE	Probabilities of killing an air base attacker escort
PKBA	Probabilities of killing a battlefield attacker
PKBD	Probabilities of killing a battlefield defender

TABLE A-1 (Cont'd)

Card Key	Input Parameters It Contains
PKBE	Probabilities of killing a battlefield attacker escort
PKFA	Probabilities of killing a forward SAM attacker
PKFS	Probabilities of killing a forward SAM
PKNS	Probabilities of killing a non-sheltered plane
PKRA	Probabilities of killing a rear SAM attacker
PKRS	Probabilities of killing a rear SAM
PKSH	Probabilities of killing a sheltered plane
REIN	Plane reinforcements by stage
RUN*	Run options and title
STAGE*	Number of stages and engagement cycles per stage
STRT**	Strategy specifications by stage
TRIAL	Initial number of planes and length of a TRIAL war
VALU	Value of an undamaged plane at end of war
WGHT	Objective function weights by stage

* At least one card required for every run.
**At least one card required for each side for every run.

ABAF Card

One ABAF card for each aircraft type is used to specify the fraction of the aircraft assigned to each mission which is vulnerable to airbase attack. By specifying different fractions, aircraft flying selected missions can be made invulnerable or partially vulnerable to the opponent's airbase attack. Generally larger fractions are more applicable for aircraft prosecuting missions close to friendly airbases (e.g., ABD, BD, CAS, and forward SAM suppression) than for aircraft flying missions deep in the opponent's territory (e.g., ABA and rear SAM suppression). Aircraft whose fractions are set equal to zero might include planes housed in impenetrable shelters or long-range bombers flying from a sanctuary base.

The ABAF card is read under

FORMAT (A4, A1, I1, 4X, 8F5.0)

and its fields are assigned as follows:

A4 - (1-4)	-	"ABAF"
A1 - (5)	-	"B" or "R" to indicate the vulnerability fractions are for either a Blue or Red aircraft type
I1 - (6)	-	the number of the aircraft type for which the fractions are applicable
8F5.0 - (11-50)	-	the fraction of the aircraft flying a particular mission which is vulnerable to airbase attack. The fraction specified in columns 11-15 corresponds to the 1st mission assigned on the associated MISS card, columns 16-20 to the 2nd, etc. Default value is 1.

CASF Card

One CASF card for each side is used to specify the firepower delivered per CAS sortie flown. The units used to specify firepower on the CAS card must be consistent with those used on both the DFRC and the DIVF cards. The CASF card is read under

FORMAT (A4, A1, 5X, 4F5.4)

and its fields are defined as follows:

A4 - (1-4)	-	"CASF"

A1 - (5)	-	"B" or "R" to indicate the firepower rates are for either the Blue or Red side
4F5.4 - (11-30)	-	the firepower delivered per CAS sortie by aircraft of types 1 thru 4 respectively. Default value is 0.

DFRC Card

One DFRC card for each side is used to specify the ground division firepower reduction produced by each CAS sortie flown. The units used to specify firepower reduction on the DFRC card must be consistent with those used on both the CASF and the DIVF cards. The DFRC card is read under

FORMAT (A4, A1, 5X, 4F5.4)

and its fields are defined as follows:

A4 - (1-4)	-	"DFRC"
A1 - (5)	-	"B" or "R" to indicate the ground division firepower reductions are produced by either a Blue or Red CAS sortie.
4F5.4 - (11-30)	-	the ground division firepower reduction resulting from a CAS sortie flown by aircraft of types 1 thru 4 respectively. Default value is 0.

DIVF Card

One DIVF card for each side is used to specify the firepower per ground division. The units used to specify firepower on the DIFV card must be consistent with those used on both the CASF and the DFRC card. The DIVF card is read under

FORMAT (A4, A1, 5X, F5.0)

and its fields are assigned as follows:

A4 - (1-4)	-	"DIVF"
A1 - (5)	-	"B" or "R" to indicate the firepower is specified for each Blue or Red ground divisions
F5.0 - (11-15)	-	the firepower per ground division. Default value is 0.

END Card

The END card signals the end of the scenario and attrition parameter cards required by ATACM1 and the beginning of the optional TRIAL cards. The END card is read under

FORMAT (A3)

and its only field contains

A3 - (1-3)	-	"END"

FEBAM Card

As many as four FEBAM cards may be used to specify the rate of FEBA movement per cycle as a function of the ratio of Blue ground firepower to Red ground firepower. Any functional relationship desired can be input by specifying up to 28 points that lie on the graph of the desired function. The model linearly interpolates between these points to yield a piecewise linear approximation. The quality of the approximation depends upon the shape of the desired graph and the number and location of points chosen. The coordinates of the points are read from the FEBAM card under

FORMAT (A5, I1, 4X, 7(2F5.0))

and the fields are assigned as follows:

A5 - (1-5)	-	"FEBAM"
I1 - (6)	-	card sequence number. Seven points (smallest firepower ratio to largest) are input on each card to a maximum of four cards or 28 points. The first card has sequence number 1, the second card (if required) has sequence number 2, etc.
7(2F5.0)-(11-80)	-	x and y coordinates of the ith point on the graph x = ratio of Blue ground firepower to Red ground firepower (always positive) y = FEBA movement per engagement cycle (positive movement corresponds to Blue advance) Default value for (x,y) is (0,0).

FINIS Card

The FINIS card signals the end of the TRIAL cards and consequently the end of the input data. The FINIS card is read under

FORMAT (A5)

and its only field contains

A5 - (1-5)	-	"FINIS"

GRID Card

One GRID card for each aircraft type is required to indicate the discrete numbers of aircraft for which optimal plays and associated bounds are explicitly computed. The GRID card is read under

FORMAT (A4, A1, I1, 4X, 11I5)

and its fields are assigned as follows:

A4 - (1-4)	-	"GRID"
A1 - (5)	-	"B" or "R" to indicate the grid levels are for either a Blue or Red aircraft type
I1 - (6)	-	the number of aircraft type for which the grid levels are applicable. Aircraft types are numbered in ascending order from 1 to 4.
11I5 - (11-65)	-	the discrete numbers of aircraft of the specified type for which optimal plays and bounds are explicitly generated. The first grid level must be 0 followed by remaining grid levels listed in ascending order. The maximum number of grid levels for an aircraft type is 11. The largest grid level should be the upper bound on the initial number of planes of this type plus the total number of reinforcements which can be added during a trial war.

MISS Card

One MISS card is required for each aircraft type to specify its minimum allocation fraction, the missions it can prosecute, and the associated sortie rates. Each MISS card is read under

FORMAT (A4, A1, I1, 4X, I2, 1X, 8(2X,I1), 3X, 8F5.2)

and the fields are assigned as follows:

A4 - (1-4)	-	"MISS"
A1 - (5)	-	"B" or "R" to indicate the mission information is for either a Blue or Red aircraft type
I1 - (6)	-	the number of the aircraft type
I2 - (11-12)	-	the denominator of the minimum allocation fraction for this aircraft type. The minimum allocation fraction is the smallest fraction of the total number of planes which can be assigned to any mission. All fractional assignments are integer multiples of the minimum allocation fraction.
8(2X,I1)-(14-37)	-	numerical codes (see Table A-2) corresponding to the missions to which planes of the specified type may be assigned. A maximum of eight missions may be specified for an aircraft type.
8F5.2-(41-80)	-	sortie rates per engagement cycle associated with the missions specified in the preceding columns. The sortie rates must be specified in the same relative order as the mission codes.

TABLE A-2

AIR MISSION CODES

Code	Mission
1	Close air support (CAS)
2	Airbase attack (ABA)
3	Battlefield defense (BD)
4	Airbase defense (ABD)
5	Close air support escort (CASE)
6	Airbase attack escort (ABAE)
7	Forward SAM suppression (FSS)
8	Rear SAM suppression (RSS)
9	No assigned mission

<u>NDIV Card</u>

One NDIV card for each side is used to specify the number of ground divisions available. The NDIV card is read under

FORMAT (A4, A1, 5X, I5)

and its fields are defined as follows:

A-4 - (1-4)	-	"NDIV"
A1 - (5)	-	"B" or "R" to indicate the number of divisions correspond to either the Blue or Red side
I5 - (11-15)	-	the number of ground divisions. Default value is 0.

<u>NSAM Card</u>

One NSAM card for each side is used to specify the number of forward and rear SAMs available. The NSAM card is read under

FORMAT (A4, A1, 5X, 2I5)

and its fields are defined as follows:

A4 - (1-4)	-	"NSAM"
A1 - (5)	-	"B" or "R" to indicate the number of SAMs correspond to either the Blue or Red side
I5 - (11-15)	-	the number of forward SAMs. Default value is 0.
I5 - (16-20)	-	the number of rear SAMs. Default value is 0.

NSHL Card

One NSHL card for each side is used to specify the number of aircraft shelters available. The NSHL card is read under

FORMAT (A4, A1, 5X, I5)

and its fields are assigned as follows:

A4 - (1-4)	-	"NSHL"
A1 - (5)	-	"B" or "R" to indicate the number of shelters correspond to either the Blue or Red side
I5 - (11-15)	-	the number of aircraft shelters. Default value is 0.

OWGHT Card

The basic form of the overall function used to generate optimal conservative strategies is given by

$$F = w_1 f_1 + w_2 f_2 + w_3 f_3 \qquad \text{(A-1)}$$

where f_1 = difference of total Blue minus total Red CAS firepower

f_2 = difference of total Blue minus total Red (CAS firepower + ground firepower)

and f_3 = total FEBA movement (positive movement corresponds to Blue advance).

One OWGHT card is used to specify the values of the weights, w_1, w_2, and w_3 to be used in computing F. The OWGHT card is read under

FORMAT (A5, 5X, 3F5.0)

and its fields are defined as follows:

A5 - (1-5)	-	"OWGHT"
F5.0 - (11-15)	-	the value of w_1. Default value is 1.
F5.0 - (16-20)	-	the value of w_2. Default value is 0.
F5.0 - (21-25)	-	the value of w_3. Default value is 0.

PKAA Card

For each aircraft type assigned an airbase attack (ABA) mission, one PKAA card is required to specify the probabilities that such an aircraft is killed by an opposing airbase defender or SAM in a one-on-one engagement. The PKAA card is read under

FORMAT (A4, A1, I1, 4X, 4F5.3, 5X, 2F5.3)

and its fields are defined as follows:

A4 - (1-4)	-	"PKAA"
A1 - (5)	-	"B" or "R" to indicate the ABA aircraft belongs to either the Blue or Red forces
I1 - (6)	-	the number of the ABA aircraft type
4F5.3-(11-30)	-	the probability the ABA aircraft is killed by opposing ABD's of types 1 thru 4 respectively. Default value is 0.
F5.3-(36-40)	-	the probability the ABA aircraft is killed by an opposing forward SAM. Default value is 0.
F5.3 - (41-45)	-	the probability the ABA aircraft is killed by an opposing rear SAM. Default value is 0.

PKAD Card

For each aircraft type assigned an airbase defense (ABD) mission, one PKAD card is required to specify the probabilities that such an aircraft is killed by an opposing airbase attacker or airbase attack escort in a one-on-one engagement. The PKAD card is read under

FORMAT (A4, A1, I1, 4X, 4F5.3, 5X, 4F5.3) and its fields are assigned as follows:

A4 - (1-4)	-	"PKAD"
A1 - (5)	-	"B" or "R" to indicate the ABD aircraft belongs to either the Blue or Red forces
I1 - (6)	-	the number of the ABD aircraft type
4F5.3- (11-30)	-	the probability the ABD aircraft is killed by opposing ABA's of types 1 thru 4 respectively. Default value is 0.
4F5.3- (36-55)	-	the probability the ABD aircraft is killed by opposing ABA escorts of types 1 thru 4 respectively. Default value is 0.

<u>PKAE Card</u>

For each aircraft type assigned an airbase attack escort (ABAE) mission, one PKAE card is required to specify the probabilities that such an aircraft is killed by an opposing airbase defender or SAM in a one-on-one engagement. The PKAE card is read under

FORMAT (A4, A1, I1, 4X, 4F5.3, 5X, 2F5.3)

and its fields are defined as follows:

A4 - (1-4)	-	"PKAE"
A1 - (5)	-	"B" or "R" to indicate the ABAE aircraft belongs to either the Blue or Red forces
I1 - (6)	-	the number of the ABAE aircraft type
4F5.3- (11-30)	-	the probability the ABAE aircraft is killed by opposing ABD's of types 1 thru 4 respectively. Default value is 0.
F5.3 - (36-40)	-	the probability the ABAE aircraft is killed by an opposing forward SAM. Default value is 0.
F5.3 - (41-45)	-	the probability the ABAE aircraft is killed by an opposing rear SAM. Default value is 0.

<u>PKBA Card</u>

For each aircraft type assigned a battlefield attack (CAS) mission, one PKBA card is required to specify the probabilities that such an aircraft is killed by an opposing battlefield defender or forward SAM in a one-on-one engagement. The PKBA card is read under

FORMAT (A4, A1, I1, 4X, 4F5.3, 5X, F5.3)

and its fields are defined as follows:

A4 - (1-4)	-	"PKBA"
A1 - (5)	-	"B" or "R" to indicate the CAS aircraft belongs to either the Blue or Red forces
I1 - (6)	-	the number of the CAS aircraft type
4F5.3- (11-30)	-	the probability the CAS aircraft is killed by opposing BD's of types 1 thru 4 respectively. Default value is 0.
F5.3- (36-40)	-	the probability the CAS aircraft is killed by an opposing forward SAM. Default value is 0.

PKBD Card

For each aircraft type assigned a battlefield defense (BD) mission, one PKBD card is required to specify the probabilities that such an aircraft is killed by an opposing battlefield attacker (CAS) or battlefield attack escort (CASE) in a one-on-one engagement. The PKBD card is read under

FORMAT (A4, A1, I1, 4X, 4F5.3, 5X, 4F5.3)

and its fields are defined as follows:

A4 - (1-4)	-	"PKBD"
A1 - (5)	-	"B" or "R" to indicate the BD aircraft belongs to either the Blue or Red forces
I1 - (6)	-	the number of the BD aircraft type
4F5.3- (11-30)	-	the probability the BD aircraft is killed by opposing CAS's of types 1 thru 4 respectively. Default value is 0.
4F5.3- (36-55)	-	the probability the BD aircraft is killed by opposing CAS escorts of types 1 thru 4 respectively. Default value is 0.

PKBE Card

For each aircraft type assigned a battlefield attack escort (CASE) mission, one PKBE card is required to specify the probabilities that such an aircraft is killed by an opposing battlefield defender or forward SAM in a one-on-one engagement. The PKBE card is read under

FORMAT (A4, A1, I1, 4X, 4F5.3, 5X, F5.3)

and its fields are defined as follows:

A4 - (1-4)	- "PKBE"
A1 - (5)	- "B" or "R" to indicate the CASE aircraft belongs to either the Blue or Red forces
I1 - (6)	- the number of the CASE aircraft type
4F5.3- (11-30)	- the probability the CASE aircraft is killed by opposing BD's of types 1 thru 4 respectively. Default value is 0.
F5.3- (36-40)	- the probability the CASE aircraft is killed by an opposing forward SAM. Default value is 0.

PKFA Card

For each aircraft type assigned a forward SAM suppression (FSS) mission, one PKFA card is required to specify the probability that such an aircraft is killed by an opposing forward SAM in a one-on-one engagement. The PKFA card is read under

FORMAT (A4, A1, I1, 4X, F5.3)

and its fields are defined as follows:

A4 - (1-4)	- "PKFA"
A1 - (5)	- "B" or "R" to indicate the FSS aircraft belongs to either the Blue or Red forces
I1 - (6)	- the number of the FSS aircraft type
F5.3- (11-15)	- the probability the FSS aircraft is killed by an opposing forward SAM. Default value is 0.

PKFS Card

One PKFS card for each side is used to specify the probabilities that a forward SAM is killed by forward SAM suppressors. The PKFS card is read under

FORMAT (A4, A1, 5X, 4F5.3)

and its fields are defined as follows:

A4 - (1-4)	- "PKFS"
A1 - (5)	- "B" or "R" to indicate the SAM being attacked belongs to either the Blue or Red forces
4F5.3 - (11-30)	- the probability the SAM is killed by opposing FSS's of types 1 thru 4 respectively. Default value is 0.

PKNS Card

One PKNS card for each side is used to specify the probabilities that a non-sheltered aircraft on the airbase is killed by airbase attackers. These kill probabilities are only applicable to the fraction specified as vulnerable to airbase attack on the ABAF card. The PKNS card is read under

FORMAT (A4, A1, 5X, 4F5.3)

and its fields are assigned as follows:

A4 - (1-4)	- "PKNS"
A1 - (5)	- "B" or "R" to indicate the airbase being attacked is either Blue or Red
4F5.3 - (11-30)	- the probability a non-sheltered, vulnerable aircraft on the airbase is killed by opposing ABA's of types 1 thru 4 respectively. Default value is 0.

PKRA Card

For each aircraft type assigned a rear SAM Suppression (RSS) mission, one PKRA card is required to specify the probabilities that such an aircraft is killed by an opposing SAM in a one-on-one engagement. The PKRA card is read under

FORMAT (A4, A1, I1, 4X, 2F5.3)

and its fields are defined as follows:

A4 - (1-4)	-	"PKRA"
A1 - (5)	-	"B" or "R" to indicate the RSS aircraft belongs to either the Blue or Red forces
I1 - (6)	-	the number of the RSS aircraft type
F5.3- (11-15)	-	the probability the RSS aircraft is killed by an opposing forward SAM. Default value is 0.
F5.3- (16-20)	-	the probability the RSS aircraft is killed by an opposing rear SAM. Default value is 0.

PKRS Card

One PKRS card for each side is used to specify the probabilities that a rear SAM is killed by rear SAM suppressors. The PKRS card is read under

FORMAT (A4, A1, 5X, 4F5.3)

and its fields are assigned as follows:

A4 - (1-4)	-	"PKRS"
A1 - (5)	-	"B" or "R" to indicate the SAM being attacked belongs to either the Blue or Red forces
4F5.3- (11-30)	-	the probability the SAM is killed by opposing RSS's of types 1 thru 4 respectively. Default value is 0.

PKSH Card

One PKSH card for each side is used to specify the probabilities that a sheltered aircraft on the airbase is killed by airbase attackers.

PKSH card is read under

FORMAT (A4, A1, 5X, 4F5.3)

and its fields are assigned as follows:

A4 - (1-4)	-	"PKSH"
A1 - (5)	-	"B" or "R" to indicate the airbase being attacked is either Blue or Red
4F5.3- (11-30)	-	the probability a sheltered, vulnerable aircraft on the airbase is killed by opposing ABA's of types 1 thru 4 respectively. Default value is 0.

REIN Card

The REIN card is used to specify numerical and/or fractional aircraft reinforcements as a function of stage. The REIN card is read under

FORMAT (A4, A1, 1X, I2, 4F5.0, 5X, 4F5.0)

and its fields are assigned as follows:

A4 - (1-4)	-	"REIN"
A1 - (5)	-	"B" or "R" to indicate the specified reinforcements are for either the Blue or Red side
I2 - (7-8)	-	the number of the stage when the reinforcements arrive. In every case reinforcement occurs at the beginning of the stage before attrition is computed.
4F5.0- (11-30)	-	the number of aircraft of types 1 thru 4 respectively added (or subtracted if entry is negative) at the beginning of the specified stage. Default value is 0.
4F5.0- (36-55)	-	the fractional increase (or decrease if entry is negative) in the number of aircraft of types 1 thru 4 respectively at the beginning of the specified stage. If both a fractional and a numerical change are specified for the same stage, the fractional change is applied before the numerical change. Default value for the fractional change is 0.

RUN Card

The RUN card, which must precede all other input cards, contains parameters to control input, execution, and output options. The RUN card is read under

FORMAT (A3, 7X, 8I1, 2X, 6A10)

and its fields are assigned as follows:

A3 - (1-3) -	"RUN"
I1 - (11) -	print option if 0 or blank input parameters are printed if 1 output is suppressed
I1 - (12) -	print option if 0 or blank the sets of pure strategies available to Blue and Red are printed if 1 output is suppressed
I1 - (13) -	print option if 0 or blank the set of discrete states corresponding to the number of Blue and Red planes available is printed if 1 output is suppressed
I1 - (14) -	abort option if 0 or blank execution proceeds to normal termination if 1 execution is terminated immediately after run-time estimates are printed
I1 - (15) -	BATTLE-TAPE option if 0 or blank one-stage battle outcomes for each state and pure strategy combination are not available on tape and must be computed. The computed battle outcomes are <u>not</u> written and stored on the BATTLE-TAPE if 1 one-stage battle outcomes for each state and pure strategy combination are not available on tape and must be computed. The computed battle outcomes are

written and stored on the BATTLE-TAPE

if 2 one-stage battle outcomes for each state and pure strategy combination are read from the BATTLE-TAPE

I1 - (16)	-	debug option
		if 0 or blank debug output is suppressed
		if 1 one-stage battle outcomes for each state and pure strategy combination are printed
I1 - (17)	-	debug option
		if 0 or blank debug output is suppressed
		if 1 MAXMIN/MINMAX plays and corresponding bounds are printed for each stage and state
I1 - (18)	-	debug option
		if 0 or blank debug output is suppressed
		if 1 beta weights used for linear interpolation are printed each time subroutine BETAS is called
6A10-(21-80)	-	optional run title

<u>STAGE Card</u>

The STAGE card is used to specify the number of stages in the campaign and the number of engagement cycles per stage. It is read under

FORMAT (A5, 5X, I2, 2X, I2)

and contains the following values:

A5 - (1-5)	-	"STAGE"
I2 - (11-12)	-	the number of stages in the campaign for which plays and associated bounds are generated (≤99). Trial wars of longer duration than the number of stages specified can not be evaluated.
I2 - (15-16)	-	the number of engagement cycles per stage (≤99).

STRT Card

The STRT card is used to specify any, all, or none of the fractional allocations of aircraft to missions as a function of stage. One STRT card is required to specify each time the allocation fractions change for either Blue or Red aircraft types. If no mission allocation fractions are specified for a given stage, ATACM1 optimizes strategy selection from the complete set of possible pure strategies; if the mission allocation fractions are specified for a subset of the missions, the model optimizes strategy selection from the associated subset of possible pure strategies; if allocation fractions are specified for all missions, the set of possible pure strategies consists of a single strategy and optimal selection reduces to the selection of this single specified strategy. Thus by using different combinations of STRT cards, ATACM1 can be used to evaluate the effects of Blue, Red, or both sides using optimal, sub-optimal, or user-specified strategies against its opponent.

The STRT card is read under

FORMAT (A4, A1, 1X, I2, 4 (2X, 8I2))

with the fields defined as follows:

A4 - (1-4)	- "STRT"
A1 - (5)	- "B" or "R" to indicate the specified allocations are for either the Blue or Red side
I2 - (7-8)	- the upper bound on the range of stages over which the allocations specified are in effect. The lower bound is 1 plus the upper bound specified on the preceding STRT card for the same side. The lower bound for the first STRT card for either side is assumed to be 1. The upper bound for the last STRT card for either side must be greater than or equal to the number of stages specified on the STAGE card.
4(2X,8I2)-(9-80)	- the fractional assignments of the ith aircraft type ($1 \leq i \leq 4$) to its missions, specified as integer multiples of the corresponding minimum allocation fraction. Columns 9-26 correspond to aircraft type 1, 27-44 to type 2, 45-62 to type 3, 63-80 to type 4. The

integer multiples for the ith aircraft type must be specified in the same order as the missions are assigned on the MISS card. To indicate that the allocation fraction for a mission is unspecified, the corresponding 2 character field must be punched with a right-justified asterisk (" *").

TRIAL Card

One TRIAL card is required for each trial war evaluation desired. If input to ATACM1, the first TRIAL card must follow the END card and the last must be followed by a FINIS card. Each TRIAL card is read under

FORMAT (A5, 5X, I2, 2X, 3I1, 3X, 4F5.0, 5X, 4F5.0)

and its fields are assigned as follows:

A5 - (1-5)	-	"TRIAL"
I2 - (11-12)	-	the number of stages in the trial war
I1 - (15)	-	print option if 0 or blank the number of planes available and the objective function value are printed for every stage if 1 output is suppressed
I1 - (16)	-	print option if 0 or blank the optimal aircraft allocation strategies for Blue and Red are printed for every stage if 1 output is suppressed
I1 - (17)	-	print option if 0 or blank the values of each of the three objective functions (f_1, f_2, and f_3 as described under the OWGHT card) are printed for every stage if 1 output is suppressed
4F5.0-(21-40)	-	the number of Blue aircraft of types 1 thru 4 respectively available at the beginning of the trial war

4F5.0-(46-65)	- the number of Red aircraft of types 1 thru 4 respectively available at the beginning of the trial war

VALU Card

One VALU card for each side is used to specify the residual value of an undamaged plane at the end of the war. The units used to specify this residual value should be consistent with those used on the CASF card. The VALU card is read under

FORMAT (A4, A1, 5X, 4F5.0)

and its fields are defined as follows:

A4 - (1-4)	- "VALU"
A1 - (5)	- "B" or "R" to indicate the values specified apply to either Blue or Red aircraft
4F5.0-(11-30)	- the residual value of an undamaged aircraft of types 1 thru 4 respectively. Default value is 0.

WGHT Card

Each component, f_j, of the overall objective function described under the OWGHT card (Equation A-1) can be expanded as

$$f_j = \sum_{t=1}^{T+1} f_{jt} \quad \text{for } j = 1, 2, 3 \qquad \text{(A-2)}$$

where

$$f_{1t} = \left\{ \begin{array}{l} \text{weighted difference of Blue minus Red CAS firepower} \\ \text{delivered during stage t} \\ b_t\, CAS_{Bt} - r_t\, CAS_{Rt} \end{array} \right. \qquad \text{(A-3)}$$

$$f_{2t} = \left\{ \begin{array}{l} \text{weighted difference of Blue minus Red total firepower} \\ \text{delivered during stage t} \\ b_t\, TFP_{Bt} - r_t\, TFP_{Rt} \end{array} \right. \qquad \text{(A-4)}$$

$$f_{3t} = \left\{ \begin{array}{l} \text{weighted FEBA movement during stage t} \\ \dfrac{b_t + r_t}{2} \quad \text{(FEBA movement during stage t)} \end{array} \right. \qquad \text{(A-5)}$$

WGHT cards are used to specify the values of the weights b_t and r_t as a function of stage. Each WGHT card is read under

FORMAT (A4, A1, 1X, I2, 2X, F5.0)

and its fields are assigned as follows:

A4 - (1-4)	-	"WGHT"
A1 - (5)	-	"B" or "R" to indicate whether the value specified is a Blue or Red weight (b_t or r_t)
I2 - (7-8)	-	the number of the stage t for which the specified weight is applicable
F5.0 - (11-15)	-	the value of the weight. Default value is 1.

As an aid to the user, Figure A-3 summarizes the formats of all the input cards described above and the default values used if a card is not supplied. With the exception of the TRIAL card, all cards for which default values are not specified are required for every run.

DATA DECK STRUCTURE

As alluded to in the descriptions of the individual cards, ATACM1 permits considerable freedom in the ordering of cards within a run deck. Table A-3 summarizes the order constraints for those card types subject to special restrictions. The only sequence requirement applicable to cards not listed in the table is that they follow the RUN card and precede the END card. Figure A-4 presents a sample run deck with the input cards in an acceptable order.

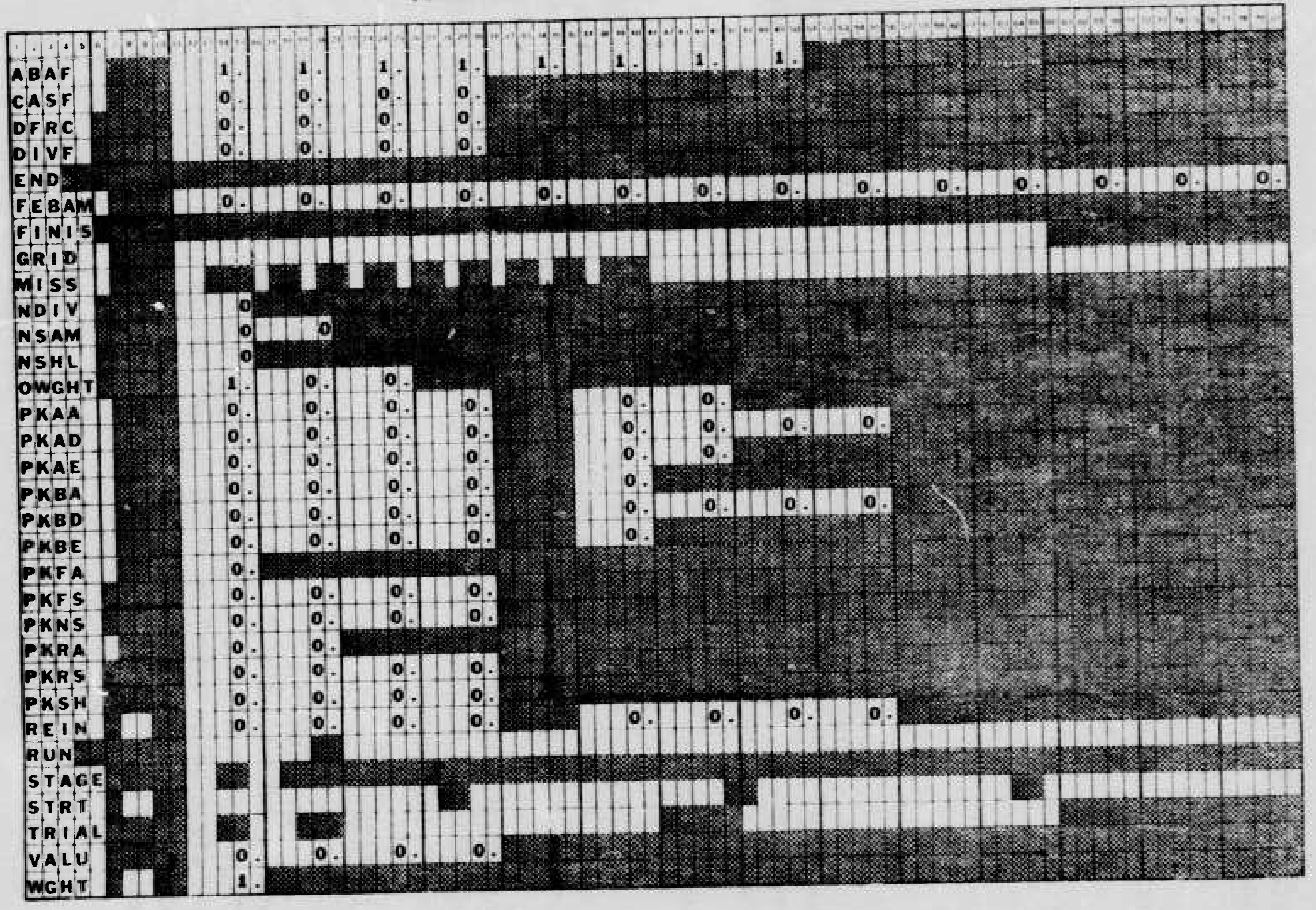
OUTPUTS

The possible outputs of ATACM1 include the printed output which details the results of the run, the TRIAL-TAPE which can be used by ATACM2 to evaluate additional trial wars, and a BATTLE-TAPE containing one-stage battle outcomes which can be used to speed the execution of certain subsequent runs of ATACM1. Each of these outputs are described below.

PAB-249

FIGURE A-3

FORMATS AND DEFAULT VALUES FOR INPUTS TO ATACM



ABAF	1.	1.	1.	1.	1.	1.	1.	1.						
CASF	0.	0.	0.	0.										
DFRC	0.	0.	0.	0.										
DIVF	0.	0.	0.	0.										
END														
FEBAM	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
FINIS														
GRID														
MISS														
NDIV	0													
NSAM	0	0												
NSHL	0													
OWGHT	1.	0.	0.											
PKAA	0.	0.	0.	0.		0.	0.							
PKAD	0.	0.	0.	0.		0.	0.	0.	0.					
PKAE	0.	0.	0.	0.		0.	0.							
PKBA	0.	0.	0.	0.		0.								
PKBD	0.	0.	0.	0.		0.	0.	0.	0.					
PKBE	0.	0.	0.	0.		0.								
PKFA	0.													
PKFS	0.	0.	0.	0.										
PKNS	0.	0.												
PKRA	0.	0.												
PKRS	0.	0.	0.	0.										
PKSH	0.	0.	0.	0.										
REIN	0.	0.	0.	0.		0.	0.	0.	0.					
RUN														
STAGE														
STRT														
TRIAL														
VALU	0.	0.	0.	0.										
WGHT	1.													

TABLE A-3

SEQUENCE RESTRICTIONS ON INPUT CARDS TO ATACM1

Card Type	Restriction
RUN	First card in the data deck
STRT	In ascending time sequence (e.g., a STRTB card for stage i must precede a STRTB card for stage j if i<j)
END	Follow scenario and attrition cards/precede the optional TRIAL and FINIS cards
TRIAL	Follow the END card/precede the FINIS card (optional)
FINIS	Follow TRIAL cards/last card in the data deck (optional)

Printed Output

Unless explicitly suppressed by the print options listed in Table A-4, every run of ATACM1 prints:

- both the input deck and the input parameters reformatted for readability
- the numbers of pure strategies available to Blue and Red
- lists of the pure strategies available to Blue and Red
- the number of possible states
- a list of the possible states
- run-time estimates

FIGURE A-4

SAMPLE RUN DECK FOR ATACM1

```
RUN                 SAMPLE RUN -- 2 BLUE AIRCRAFT TYPES, 1 RED AIRCRAFT TYPE
STAGE      2  1
NDIVB        10
NDIVR        15
DIVFB         9.
DIVFR         7.
GRIDB1        0  333  667 1000
GRIDB2        0  200  400
GRIDR1        0  400  800 1200
MISSB1     2  1  2  7  8                     2.0  1.0  2.0  1.0
MISSB2     3  3  4  5  6                     2.0  2.0  2.0  1.0
MISSR1     2  1  2  3  4                     2.0  1.0  2.0  2.0
STRTB  2   1 * * *          * * 1 *
STRTR  1   0 2 0 0
STRTR  2   * * * *
REINB  2    100                                .2
REINR  2    200
VALUB         8.   0.
VALUR         3.
CASFB         2.
CASFR         3.
OWGHT         0.  1.0   0.
NSHLB       100
NSHLR       300
NSAMB        30   50
NSAMR        20   40
ABAFB1      1.0   .5  1.0   .5
ABAFB2      1.0  1.0  1.0   .5
ABAFR1       .5   .5   .5   .5
WGHTB   1    .5
PKSHB       .20
PKSHR       .25
PKNSB       .35
PKNSR       .50
PKRSR       .06
PKFSR       .08
DFRCB        .5
DFRCR        .8
FEBAM1       0.  -1.  .99  -1. 1.01    1. 200.   1.
PKBAB1      .10                       .16
PKBAR1           .12                  .17
PKAAB1      .09                       .15  .20
PKAAR1           .11                  .14  .21
PKBDB2      .05
PKBDR1      .04                            .12
PKADB2      .03
PKADR1      .02                            .03
PKBEB2      .05                       .03
PKAEB2      .07                       .01  .02
PKFAB1      .01
PKRAB1      .01  .03
END
TRIAL       2          400  100                   500
TRIAL       1  11      600  200                   600
TRIAL       2    1     800  300                   800
FINIS
```

TABLE A-4

RUN CARD
PRINT OPTION CONTROLS

To suppress the print of:	Punch a 1 in RUN card column
Input parameters	11
All possible strategies for Blue and Red	12
All possible states	13

If trial war evaluations are requested during the execution of ATACM1, additional outputs are printed under the control of the print option parameters on the TRIAL card. Specifically, unless explicitly suppressed by the print options listed in Table A-5, every trial war evaluation prints:

- the TRIAL card parameters, the MAXMIN and MINMAX bounds on the objective function, and the value of the objective function produced by playing Blue's MAXMIN strategy against Red's MINMAX strategy
- the number of planes available on both sides and the value of the objective function as a function of stage
- the optimal aircraft allocation strategies as a function of stage
- the individual values of the three objective functions (i.e., Blue-Red CAS Firepower, Blue-Red Total Firepower, and FEBA movement) listed as a function of stage

Figures A-5 and A-6, which were produced by the sample input deck of Figure A-4, illustrate the outputs printed under the control of the RUN and TRIAL cards respectively.

TABLE A-5

TRIAL CARD
PRINT OPTION CONTROLS

To suppress the print (by stage) of:	Punch a 1 in TRIAL card column
Number of planes/objective function value	15
Optimal strategies for Blue and Red	16
All three objective function values	17

BATTLE-TAPE

The BATTLE-TAPE is an optional output of ATACM1 which contains the one-stage battle assessments computed in the initialization phase of the run (see Figure A-2). The BATTLE-TAPE has two potential applications:

- it provides a partial backup/restart capability should an ATACM1 job be terminated abnormally after the battle assessments are computed, and

- it can be used to input rather than recompute battle assessments for perturbation runs of ATACM1 in which those parameters affecting one-stage battle outcomes are unchanged.

The advisability of specifying a BATTLE-TAPE as an output of a long-running ATACM1 job should be obvious. If such a job aborts abnormally after the BATTLE-TAPE is written, a rerun can be made by simply assigning the BATTLE-TAPE as an input, punching a "2" in column 15 of the RUN card, and resubmitting the job. The resubmitted job reads the battle assessments directly from the BATTLE-TAPE, thus saving the time required for their calculation.

In the case of a shorter run, the decision whether to create a BATTLE-TAPE depends upon how applicable the computed battle assessments will be to other related runs of ATACM1. Table A-6 lists those input parameters which can be changed without affecting one-stage battle outcomes and thus defines the set of related runs which may share the

PAB-249

FIGURE A-5

SAMPLE OUTPUT PRINTED UNDER CONTROL OF RUN CARD

```
SAMPLE RUN -- 2 BLUE AIRCRAFT TYPES. 1 RED AIRCRAFT TYPE

NUMBER OF STAGES =  2
NUMBER OF CYCLES PER STAGE =  1
NUMBER OF BLUE PLANE TYPES = 2
NUMBER OF RED  PLANE TYPES = 1
NUMBER OF BLUE DIVISIONS =      10
NUMBER OF RED  DIVISIONS =      15
FIREPOWER PER BLUE DIVISION =      9.0000
FIREPOWER PER RED  DIVISION =      7.0000
NUMBER OF BLUE SHELTERS =    100
NUMBER OF RED  SHELTERS =    300
NUMBER OF BLUE FORWARD SAMS =      30
NUMBER OF RED  FORWARD SAMS =      20
NUMBER OF BLUE REAR SAMS =      50
NUMBER OF RED  REAR SAMS =      40
OBJECTIVE FUNCTION WEIGHTS = CAS GRD -  0.00   TOTL FP -  1.00   FEBA -  0.00


                     MISSIONS ASSIGNED AND
                    ASSOCIATED SORTIE RATES


        BLUE PLANE TYPE                          RED PLANE TYPE
    1         2         3         4          1         2         3         4

 1- 2.00 3- 2.00 0- 0.00 0- 0.00     1- 2.00 0- 0.00 0- 0.00 0- 0.00
 2- 1.00 4- 2.00 0- 0.00 0- 0.00     2- 1.00 0- 0.00 0- 0.00 0- 0.00
 7- 2.00 5- 2.00 0- 0.00 0- 0.00     3- 2.00 0- 0.00 0- 0.00 0- 0.00
 8- 1.00 6- 1.00 0- 0.00 0- 0.00     4- 2.00 0- 0.00 0- 0.00 0- 0.00


                  MINIMUM ALLOCATION FRACTIONS


        BLUE PLANE TYPE                          RED PLANE TYPE
    1         2         3         4          1         2         3         4

  .50       .33      0.00      0.00        .50      0.00      0.00      0.00


                          GRID POINTS


        BLUE PLANE TYPE                          RED PLANE TYPE
    1         2         3         4          1         2         3         4

    0         0         0         0          0         0         0         0
  333       200         0         0        400         0         0         0
  667       400         0         0        800         0         0         0
 1000        -1         0         0       1200         0         0         0
```

```
                       CAS FIREPOWER PER SORTIE


          BLUE PLANE TYPE                        RED PLANE TYPE
   1         2         3         4          1         2         3         4

2.0000   -.0000   -.0000   -.0000     3.0000   -.0000   -.0000   -.0000


                   DIVISION FIREPOWER REDUCTION
                         PER CAS SORTIE

          BLUE PLANE TYPE                        RED PLANE TYPE
   1         2         3         4          1         2         3         4

 .5000   -.0000   -.0000   -.0000      .8000   -.0000   -.0000   -.0000


                      RESIDUAL VALUE OF AN
               UNDAMAGED PLANE AT END OF THE WAR


          BLUE PLANE TYPE                        RED PLANE TYPE
   1         2         3         4          1         2         3         4

8.0000   0.0000   -.0000   -.0000     3.0000   -.0000   -.0000   -.0000


                   FRACTION VULNERABLE TO ABA
                           BY MISSION


          BLUE PLANE TYPE                        RED PLANE TYPE
   1         2         3         4          1         2         3         4

1-1.000 3-1.000 0-1.000 0-1.000     1- .500 0-1.000 0-1.000 0-1.000
2- .500 4-1.000 0-1.000 0-1.000     2- .500 0-1.000 0-1.000 0-1.000
7-1.000 5-1.000 0-1.000 0-1.000     3- .500 0-1.000 0-1.000 0-1.000
8- .500 6- .500 0-1.000 0-1.000     4- .500 0-1.000 0-1.000 0-1.000


                          FEBA FUNCTION


F RATIO          0.000      .999     1.010   200.000     -.000     -.000     -.000
MOVEMENT        -1.000    -1.000     1.000     1.000     -.000     -.000     -.000
```

PAB 249

FIGURE A-5 (cont'd)

```
   OBJECTIVE              REINFORCEMENTS                                     REINFORCEMENTS
    WEIGHTS                   NUMBER                                            FRACTION
                   BLUE PLANE TYPE          RED PLANE TYPE          BLUE PLANE TYPE            RED PLANE TYPE
STAGE  BLUE   RED    1    2    3    4      1    2    3    4      1     2     3     4       1     2     3     4

  1    .50   1.00    0    0    0    0      0    0    0    0    0.00  0.00  0.00  0.00    0.00  0.00  0.00  0.00
  2   1.00   1.00  100    0    0    0    200    0    0    0    0.00   .20  0.00  0.00    0.00  0.00  0.00  0.00


                                      KILL PROBABILITIES


                              ABA AGAINST NON-SHELTERED AIRCRAFT

                   BLUE KILLS RED                          RED KILLS BLUE

                      RED TYPE                                RED TYPE
              1      2      3      4                1      2      3      4
      1     .500  -.000  -.000  -.000             .350   .350   .350   .350
      2     .500  -.000  -.000  -.000            -.000  -.000  -.000  -.000
      3     .500  -.000  -.000  -.000            -.000  -.000  -.000  -.000
      4     .500  -.000  -.000  -.000            -.000  -.000  -.000  -.000


                                ABA AGAINST SHELTERED AIRCRAFT

                   BLUE KILLS RED                          RED KILLS BLUE

                      RED TYPE                                RED TYPE
              1      2      3      4                1      2      3      4
      1     .250  -.000  -.000  -.000             .200   .200   .200   .200
      2     .250  -.000  -.000  -.000            -.000  -.000  -.000  -.000
      3     .250  -.000  -.000  -.000            -.000  -.000  -.000  -.000
      4     .250  -.000  -.000  -.000            -.000  -.000  -.000  -.000
```

FIGURE A-5 (cont'd)

```
                         KILL PROBABILITIES


                       CAS ESCORT AGAINST BD

         BLUE KILLS RED                          RED KILLS BLUE

            RED TYPE                                RED TYPE
      1        2       3       4           1       2       3       4
1   -.000   0.000   0.000   0.000        0.000   0.000   0.000   0.000
2    .120   0.000   0.000   0.000        -.000   -.000   -.000   -.000
3   -.000   0.000   0.000   0.000        0.000   0.000   0.000   0.000
4   -.000   0.000   0.000   0.000        0.000   0.000   0.000   0.000

                       BD AGAINST CAS ESCORT

         BLUE KILLS RED                          RED KILLS BLUE

            RED TYPE                                RED TYPE
      1        2       3       4           1       2       3       4
1   0.000   0.000   0.000   0.000        0.000   0.000   0.000   0.000
2   0.000   0.000   0.000   0.000         .050   -.000   -.000   -.000
3   0.000   0.000   0.000   0.000        0.000   0.000   0.000   0.000
4   0.000   0.000   0.000   0.000        0.000   0.000   0.000   0.000

                          CAS AGAINST BD

         BLUE KILLS RED                          RED KILLS BLUE

            RED TYPE                                RED TYPE
      1        2       3       4           1       2       3       4
1    .040   0.000   0.000   0.000        0.000   0.000   0.000   0.000
2   -.000   0.000   0.000   0.000         .050   -.000   -.000   -.000
3   -.000   0.000   0.000   0.000        0.000   0.000   0.000   0.000
4   -.000   0.000   0.000   0.000        0.000   0.000   0.000   0.000

                          BD AGAINST CAS

         BLUE KILLS RED                          RED KILLS BLUE

            RED TYPE                                RED TYPE
      1        2       3       4           1       2       3       4
1   -.000   0.000   0.000   0.000         .100   -.000   -.000   -.000
2    .120   0.000   0.000   0.000        0.000   0.000   0.000   0.000
3   -.000   0.000   0.000   0.000        0.000   0.000   0.000   0.000
4   -.000   0.000   0.000   0.000        0.000   0.000   0.000   0.000
```

```
                         KILL PROBABILITIES


                       ABA ESCORT AGAINST ABD

         BLUE KILLS RED                          RED KILLS BLUE

            RED TYPE                                RED TYPE
      1        2       3       4           1       2       3       4
1   -.000   0.000   0.000   0.000        0.000   0.000   0.000   0.000
2    .030   0.000   0.000   0.000        -.000   -.000   -.000   -.000
3   -.000   0.000   0.000   0.000        0.000   0.000   0.000   0.000
4   -.000   0.000   0.000   0.000        0.000   0.000   0.000   0.000

                       ABD AGAINST ABA ESCORT

         BLUE KILLS RED                          RED KILLS BLUE

            RED TYPE                                RED TYPE
      1        2       3       4           1       2       3       4
1   0.000   0.000   0.000   0.000        0.000   0.000   0.000   0.000
2   0.000   0.000   0.000   0.000         .070   -.000   -.000   -.000
3   0.000   0.000   0.000   0.000        0.000   0.000   0.000   0.000
4   0.000   0.000   0.000   0.000        0.000   0.000   0.000   0.000

                          ABA AGAINST ABD

         BLUE KILLS RED                          RED KILLS BLUE

            RED TYPE                                RED TYPE
      1        2       3       4           1       2       3       4
1    .020   0.000   0.000   0.000        0.000   0.000   0.000   0.000
2   -.000   0.000   0.000   0.000         .030   -.000   -.000   -.000
3   -.000   0.000   0.000   0.000        0.000   0.000   0.000   0.000
4   -.000   0.000   0.000   0.000        0.000   0.000   0.000   0.000

                          ABD AGAINST ABA

         BLUE KILLS RED                          RED KILLS BLUE

            RED TYPE                                RED TYPE
      1        2       3       4           1       2       3       4
1   -.000   0.000   0.000   0.000         .090   -.000   -.000   -.000
2    .110   0.000   0.000   0.000        0.000   0.000   0.000   0.000
3   -.000   0.000   0.000   0.000        0.000   0.000   0.000   0.000
4   -.000   0.000   0.000   0.000        0.000   0.000   0.000   0.000
```

PAB-249

FIGURE A-5 (cont'd)

```
                        KILL PROBABILITIES


                          SAM AGAINST CAS

                 BLUE KILLS RED                   RED KILLS BLUE

                    RED TYPE                        BLUE TYPE
             1      2      3      4           1      2      3      4
FORWARD    .170  0.000  0.000  0.000        .160  0.000  0.000  0.000

                       SAM AGAINST CAS ESCORT

                 BLUE KILLS RED                   RED KILLS BLUE

                    RED TYPE                        BLUE TYPE
             1      2      3      4           1      2      3      4
FORWARD   0.000  0.000  0.000  0.000       0.000   .030  0.000  0.000

                          SAM AGAINST AAA

                 BLUE KILLS RED                   RED KILLS BLUE

                    RED TYPE                        BLUE TYPE
             1      2      3      4           1      2      3      4
FORWARD    .140  0.000  0.000  0.000        .150  0.000  0.000  0.000
REAR       .210  0.000  0.000  0.000        .200  0.000  0.000  0.000

                       SAM AGAINST AAA ESCORT

                 BLUE KILLS RED                   RED KILLS BLUE

                    RED TYPE                        BLUE TYPE
             1      2      3      4           1      2      3      4
FORWARD   0.000  0.000  0.000  0.000       0.000   .010  0.000  0.000
REAR      0.000  0.000  0.000  0.000       0.000   .020  0.000  0.000


                        KILL PROBABILITIES


              FORWARD SAM SUPPRESSOR AGAINST SAM

                 BLUE KILLS RED                   RED KILLS BLUE

                    BLUE TYPE                       RED TYPE
             1      2      3      4           1      2      3      4
FORWARD    .080  -.000  -.000  -.000       0.000  0.000  0.000  0.000

              SAM AGAINST FORWARD SAM SUPPRESSOR

                 BLUE KILLS RED                   RED KILLS BLUE

                    RED TYPE                        BLUE TYPE
             1      2      3      4           1      2      3      4
FORWARD   0.000  0.000  0.000  0.000        .010  0.000  0.000  0.000

                REAR SAM SUPPRESSOR AGAINST SAM

                 BLUE KILLS RED                   RED KILLS BLUE

                    BLUE TYPE                       RED TYPE
             1      2      3      4           1      2      3      4
REAR       .060  -.000  -.000  -.000       0.000  0.000  0.000  0.000

               SAM AGAINST REAR SAM SUPPRESSOR

                 BLUE KILLS RED                   RED KILLS BLUE

                    RED TYPE                        BLUE TYPE
             1      2      3      4           1      2      3      4
FORWARD   0.000  0.000  0.000  0.000        .010  0.000  0.000  0.000
REAR      0.000  0.000  0.000  0.000        .030  0.000  0.000  0.000
```

FIGURE A-5 (cont'd)

```
NUMBER OF BLUE PURE STRATEGIES EQUALS    18

NUMBER OF RED  PURE STRATEGIES EQUALS    11


                  BLUE PURE STRATEGIES

STRAT    LAST             PLANE TYPE/MISSION
NUMBER   STAGE      1/1  1/2  1/7  1/8  2/3  2/4  2/5  2/6

   1       2        .50 0.00 0.00  .50 0.00 0.00  .33  .67
   2       2        .50 0.00 0.00  .50 0.00  .33  .33  .33
   3       2        .50 0.00 0.00  .50 0.00  .67  .33 0.00
   4       2        .50 0.00 0.00  .50  .33 0.00  .33  .33
   5       2        .50 0.00 0.00  .50  .33  .33  .33 0.00
   6       2        .50 0.00 0.00  .50  .67 0.00  .33 0.00
   7       2        .50 0.00  .50 0.00 0.00 0.00  .33  .67
   8       2        .50 0.00  .50 0.00 0.00  .33  .33  .33
   9       2        .50 0.00  .50 0.00 0.00  .67  .33 0.00
  10       2        .50 0.00  .50 0.00  .33 0.00  .33  .33
  11       2        .50 0.00  .50 0.00  .33  .33  .33 0.00
  12       2        .50 0.00  .50 0.00  .67 0.00  .33 0.00
  13       2        .50  .50 0.00 0.00 0.00 0.00  .33  .67
  14       2        .50  .50 0.00 0.00 0.00  .33  .33  .33
  15       2        .50  .50 0.00 0.00 0.00  .67  .33 0.00
  16       2        .50  .50 0.00 0.00  .33 0.00  .33  .33
  17       2        .50  .50 0.00 0.00  .33  .33  .33 0.00
  18       2        .50  .50 0.00 0.00  .67 0.00  .33 0.00


                  RED  PURE STRATEGIES

STRAT    LAST             PLANE TYPE/MISSION
NUMBER   STAGE      1/1  1/2  1/3  1/4

   1       1       0.00 1.00 0.00 0.00
   2       2       0.00 0.00 0.00 1.00
   3       2       0.00 0.00  .50  .50
   4       2       0.00 0.00 1.00 0.00
   5       2       0.00  .50 0.00  .50
   6       2       0.00  .50  .50 0.00
   7       2       0.00 1.00 0.00 0.00
   8       2        .50 0.00 0.00  .50
   9       2        .50 0.00  .50 0.00
  10       2        .50  .50 0.00 0.00
  11       2       1.00 0.00 0.00 0.00
```

```
NUMBER OF STATES EQUALS      48


              LIST OF ALL POSSIBLE STATES

STATE             BLUE                          RED
NUMBER      1      2     3     4         1     2     3     4

   1        0      0     0     0         0     0     0     0
   2        0      0     0     0       400     0     0     0
   3        0      0     0     0       800     0     0     0
   4        0      0     0     0      1200     0     0     0
   5        0    200     0     0         0     0     0     0
   6        0    200     0     0       400     0     0     0
   7        0    200     0     0       800     0     0     0
   8        0    200     0     0      1200     0     0     0
   9        0    400     0     0         0     0     0     0
  10        0    400     0     0       400     0     0     0
  11        0    400     0     0       800     0     0     0
  12        0    400     0     0      1200     0     0     0
  13      333      0     0     0         0     0     0     0
  14      333      0     0     0       400     0     0     0
  15      333      0     0     0       800     0     0     0
  16      333      0     0     0      1200     0     0     0
  17      333    200     0     0         0     0     0     0
  18      333    200     0     0       400     0     0     0
  19      333    200     0     0       800     0     0     0
  20      333    200     0     0      1200     0     0     0
  21      333    400     0     0         0     0     0     0
  22      333    400     0     0       400     0     0     0
  23      333    400     0     0       800     0     0     0
  24      333    400     0     0      1200     0     0     0
  25      667      0     0     0         0     0     0     0
  26      667      0     0     0       400     0     0     0
  27      667      0     0     0       800     0     0     0
  28      667      0     0     0      1200     0     0     0
  29      667    200     0     0         0     0     0     0
  30      667    200     0     0       400     0     0     0
  31      667    200     0     0       800     0     0     0
  32      667    200     0     0      1200     0     0     0
  33      667    400     0     0         0     0     0     0
  34      667    400     0     0       400     0     0     0
  35      667    400     0     0       800     0     0     0
  36      667    400     0     0      1200     0     0     0
  37     1000      0     0     0         0     0     0     0
  38     1000      0     0     0       400     0     0     0
  39     1000      0     0     0       800     0     0     0
  40     1000      0     0     0      1200     0     0     0
  41     1000    200     0     0         0     0     0     0
  42     1000    200     0     0       400     0     0     0
  43     1000    200     0     0       800     0     0     0
  44     1000    200     0     0      1200     0     0     0
  45     1000    400     0     0         0     0     0     0
  46     1000    400     0     0       400     0     0     0
  47     1000    400     0     0       800     0     0     0
  48     1000    400     0     0      1200     0     0     0
```

FIGURE A-5 (cont'd)

```
CDC 6600 TIME ESTIMATES FOR CURRENT RUN (SECONDS)

          FUNCTION          CPU TIME   I/O TIME

          SETUP                  2.0        2.0
          BATTLES               19.0        5.8
          GAMES                  7.4       11.5

          TOTAL                 28.4       19.3
```

FIGURE A-6

SAMPLE OUTPUT PRINTED
UNDER CONTROL OF TRIAL CARDS

TRIAL CARD #1

```
           NUMBER        NUMBER OF PLANES AVAILABLE                                          MAXMIN
  TRIAL      OF      ----- BLUE -----        ----- RED ------      BLUE        RED          VS
  NUMBER  STAGES    1     2     3     4     1     2     3     4    MAXMIN      MINMAX       MINMAX

     1       2     400   100    0     0   500     0     0     0   -10515       9735        -2837
```

```
                          TRIAL NUMBER     1

                NUMBER OF BLUE          NUMBER OF RED          MAXMIN
  STAGE        PLANES AVAILABLE        PLANES AVAILABLE          VS
  NUMBER       1     2     3     4     1     2     3     4     MINMAX      TOTAL

     1       297    77     0     0   481     0     0     0       442        442
     2       396    91     0     0   679     0     0     0     -3279      -2837
```

```
                          TRIAL NUMBER     1

                   PLANE ALLOCATION FRACTIONS FOR BLUE

   STAGE                   PLANE TYPE/MISSION
   NUMBER      1/1  1/2  1/7  1/8  2/3  2/4  2/5  2/6

     1         .50 0.00 0.00  .50 0.00 0.00  .33  .67
     2         .50 0.00  .50 0.00 0.00 0.00  .33  .67
```

```
                          TRIAL NUMBER     1

                   PLANE ALLOCATION FRACTIONS FOR RED

   STAGE                   PLANE TYPE/MISSION
   NUMBER      1/1  1/2  1/3  1/4

     1        0.00 1.00 0.00 0.00
     2        1.00 0.00 0.00 0.00
```

FIGURE A-6 (cont'd)

TRIAL CARD #1 (cont'd)

TRIAL NUMBER 1

STAGE NUMBER	BLUE-RED CAS FIREPOWER	TOTAL	BLUE-RED GRND+AIR FIREPOWER	TOTAL	FEBA MOVEMENT	TOTAL
1	397	397	442	442	1	1
2	-3279	-2882	-3279	-2837	0	1

TRIAL CARD #2

TRIAL NUMBER	NUMBER OF STAGES	NUMBER OF PLANES AVAILABLE BLUE 1	BLUE 2	BLUE 3	BLUE 4	RED 1	RED 2	RED 3	RED 4	BLUE MAXMIN	RED MINMAX	MAXMIN VS MINMAX
2	1	600	200	0	0	600	0	0	0	-5721	5321	-15

TRIAL NUMBER 2

STAGE NUMBER	BLUE-RED CAS FIREPOWER	TOTAL	BLUE-RED GRND+AIR FIREPOWER	TOTAL	FEBA MOVEMENT	TOTAL
1	0	0	-15	-15	-1	-1

FIGURE A-6 (cont'd)

TRIAL CARD #3

TRIAL NUMBER	NUMBER OF STAGES	BLUE 1	BLUE 2	BLUE 3	BLUE 4	RED 1	RED 2	RED 3	RED 4	BLUE MAXMIN	RED MINMAX	MAXMIN VS MINMAX
3	2	800	300	0	0	800	0	0	0	-9412	6660	-1909

TRIAL NUMBER 3

STAGE NUMBER	NUMBER OF BLUE PLANES AVAILABLE 1	2	3	4	NUMBER OF RED PLANES AVAILABLE 1	2	3	4	MAXMIN VS MINMAX	TOTAL
1	717	254	0	0	737	0	0	0	842	842
2	816	295	0	0	910	0	0	0	-2751	-1909

TRIAL NUMBER 3

PLANE ALLOCATION FRACTIONS FOR BLUE

STAGE NUMBER	PLANE TYPE/MISSION 1/1	1/2	1/7	1/8	2/3	2/4	2/5	2/6
1	.50	0.00	0.00	.50	0.00	.67	.33	0.00
2	.50	0.00	.50	0.00	.67	0.00	.33	0.00

TRIAL NUMBER 3

PLANE ALLOCATION FRACTIONS FOR RED

STAGE NUMBER	PLANE TYPE/MISSION 1/1	1/2	1/3	1/4
1	0.00	1.00	0.00	0.00
2	1.00	0.00	0.00	0.00

same BATTLE-TAPE. For example, to study the sensitivity of war outcomes to reinforcement policies, numerous runs of ATACM1 would be required in which only the reinforcement parameters on the REIN cards would change. In such a case, since reinforcement parameters are listed in Table A-6, the BATTLE-TAPE produced by the first run of ATACM1 could be used to input, rather than recompute, the one-stage battle outcomes required by the remaining runs. The only changes required in the data deck for one of these perturbation runs would be the changed REIN cards and a modified RUN card with a "2" punched in column 15.

In the CDC 6600 version of ATACM1, the BATTLE-TAPE is assigned the logical file name "TAPE10".

TABLE A-6

INPUT PARAMETERS WHICH DO NOT AFFECT ONE-STAGE BATTLE ASSESSMENTS

Card Type	Parameters	Columns
REIN	Reinforcements	11-30, 36-55
STAGE	Number of stages	11-12
VALU	Residual value of undamaged plane	11-30
WGHT	Objective function weights by stage	11-15

TRIAL-TAPE

The TRIAL-TAPE produced by ATACM1 contains the values of the major COMMON areas assigned during the execution of ATACM1 as well as the optimal strategies and associated bounds generated for each state and stage of the campaign. The tape is used exclusively by ATACM2 to input those parameters required to evaluate trial wars of varying length initiated with differing numbers of aircraft available to the opposing forces.

In the CDC 6600 version of ATACM1, the TRIAL-TAPE is assigned the logical file name "TAPE4".

EXECUTION TIME

One of the most important considerations in the use of ATACM1 is the execution time required to generate the results written on the TRIAL-TAPE. There are three phases of calculation necessary for the generation of this tape:

- SETUP - the generation of strategies and states

- BATTLES - the calculation and output to scratch disk (and BATTLE-TAPE if requested) of one-stage battle assessments for each Blue-Red strategy combination

- GAMES - the calculation of the MAXMIN/MINMAX objective function bounds and plays for each stage and state

The time required for SETUP is usually insignificant (2 seconds) compared to the times for BATTLES and GAMES. Run times for these last two phases can range from a few seconds for a simple scenario to several minutes or even hours for the most ambitious requests.

Time Estimates

The run time associated with BATTLES and GAMES is a relatively complex function of the total number of aircraft types on both sides, the number of Blue and Red pure strategies, the size of the state space, and the number of stages and cycles-per-stage in the air campaign being simulated. To provide the user with estimates of the CPU and IO times required for BATTLES and GAMES, all of these variables were incorporated into empirical formulas derived from numerous test runs made on the CDC 6600. The formulas were coded into the subroutine TIMER which prints time estimates for BATTLE and GAMES before the calculations are performed.

To use the time estimates produced by TIMER to assess the reasonableness of a particular run before it is submitted for complete execution, a preliminary run using the candidate data deck should be submitted with a 1 punched in column 14 of the RUN card. All outputs shown in Figure A-5 through the printing of the time estimates are generated in their normal manner. However, execution is terminated just before the BATTLES and GAMES phases of computation begin. The output which is generated permits verification of the accuracy of the input data and provides a basis for determining whether the length of time required for a complete run is acceptable. Because of the minimal effort and cost associated with such a data-and-time check, its use is strongly recommended for all but the simplest runs.

Use of the BATTLE-TAPE

Another consideration related to the execution time of ATACM1 is the use of the BATTLE-TAPE. As described under the outputs of ATACM1, use of the BATTLE-TAPE permits the one-stage battle assessments produced during the BATTLES phase of calculation to be written and stored on magnetic tape for use in subsequent related runs of ATACM1. Since the time required for the calculation of these battle assessments is typically a significant fraction of the total execution time, judicious job sequencing which permits the use of a single BATTLE-TAPE for numerous runs of ATACM1 will produce substantial savings in the total computer time used.

DIAGNOSTIC MESSAGES

Diagnostic messages generated by ATACM1 are classified in order of increasing severity as

INFO:	Information only -- anomaly is ignored.
ERROR:	Processing error -- job is aborted at the end of the current subroutine.
ABORT:	Abort -- job is aborted immediately.

Messages which may be produced by ATACM1 are listed and interpreted in Table A-7 in approximately the same order they are encountered during program execution.

TABLE A-7

DIAGNOSTIC MESSAGES GENERATED BY ATACM1

Message	Severity	Interpretation
DATA CARD KEY NOT RECOGNIZED	ERROR	Input card key does not match any of those listed in Table A-1, or a TRIAL or FINIS card precedes the END card.
SMALLEST GRID LEVEL MUST BE ZERO	ERROR	One of the GRID cards does not have a 0 punched in columns 11-15.
SUM OF SPECIFIED ALLOCATIONS EXCEEDS 1.0	ABORT	The sum of the allocations of an aircraft type to its missions, as specified on a STRT card, exceeds the denominator of the minimum allocation fraction for the corresponding aircraft type.
IF ALL ALLOCATIONS ARE SPECIFIED THEY MUST SUM TO 1.0	ABORT	If all the allocations of an aircraft type to its missions are specified on a STRT card, they must sum to the denominator of the minimum allocation fraction for the corresponding aircraft type.
STRATEGIES NOT SPECIFIED THRU THE LAST STAGE OF CAMPAIGN	ABORT	The number in columns 7-8 of last STRT card read for either Blue or Red was less than the total number of stages specified on the STAGE card.

TABLE A-7 (Cont'd)

Message	Severity	Interpretation
TOO MANY PURE STRATEGIES TO BE STORED IN CORE	ABORT	Either the number of pure strategies for one side exceeds 500, the number of decision vectors for a single aircraft type exceeds 200, or the total area required to store the pure strategies for both sides exceeds 25,000 words. Possible remedies include specification of fewer missions, fewer aircraft types, or a larger minimum allocation fraction.
TOO MANY STATES TO STORE VALUES AND PLAYS	ABORT	The total number of states multiplied by 10 plus the dimensioned area used to store the pure strategies exceeds 25,000 words. Either the number of states or the total number of pure strategies must be reduced.
NOT ENOUGH COMMON AREA TO STORE GAME MATRIX	ABORT	Two times the number of elements in the one-stage game matrix, plus 10 times the number of states, plus the dimensioned area used to store the pure strategies exceeds 25,000 words. Either the number of states or the number of pure strategies must be reduced.
BUFFER OUT ERROR TO TAPE9 IN SUBROUTINE READIN	ABORT	System write failure was encountered while writing STRT parameters on scratch disk (TAPE9). Rerun job or consult systems analyst.

TABLE A-7 (Cont'd)

Message	Severity	Interpretation
BUFFER IN ERROR FROM TAPE9 IN SUBROUTINE INIT	ABORT	Either a parity error or system read failure was encountered while reading STRT parameters from scratch disk (TAPE9). Rerun job or consult systems analyst.
USER REQUESTED ONLY A DATA CHECK AND TIME ESTIMATE	ABORT	A 1 was punched in column 14 of the RUN card. Execution is terminated immediately after run-time estimates are printed.
I/O ERROR ON TAPE10 IN SUBROUTINE INIT	ABORT	Either a read parity error or a system I/O failure was encountered on the BATTLE-TAPE (TAPE10). Rerun job or consult systems analyst.
BUFFER OUT ERROR TO TAPE1 IN SUBROUTINE INIT	ABORT	System write failure was encountered while writing battle assessments on scratch disk (TAPE1). Rerun job or consult systems analyst.
BUFFER IN ERROR FROM TAPE1 IN SUBROUTINE GAMES	ABORT	Either a parity error or system read failure was encountered while reading battle assessments from scratch disk (TAPE1). Rerun job or consult systems analyst.
BUFFER OUT ERROR TO TAPE4 IN MAIN PROGRAM	ABORT	System write failure was encountered while writing TRIAL-TAPE (TAPE4). Rerun job or consult systems analyst.

TABLE A-7 (Cont'd)

Message	Severity	Interpretation
BUFFER OUT ERROR TO TAPE7 IN MAIN PROGRAM	ABORT	System write failure was encountered while writing MAXMIN/MINMAX objective function values on scratch disk (TAPE7). Rerun job or consult systems analyst.
BUFFER IN ERROR FROM TAPE7 IN SUBROUTINE TRIALS	ABORT	Parity error or system read failure was encountered while reading MAXMIN/MINMAX objective function values from scratch disk (TAPE7). Rerun job or consult systems analyst.
BUFFER OUT ERROR TO TAPE8 IN MAIN PROGRAM	ABORT	System write failure was encountered while writing MAXMIN/MINMAX plays on scratch disk (TAPE8). Rerun job or consult systems analyst.
BUFFER IN ERROR FROM TAPE8 IN SUBROUTINE TRIALS	ABORT	Parity error or system read failure was encountered while reading MAXMIN/MINMAX plays from scratch disk (TAPE8). Rerun job or consult systems analyst.
DISK I/O ERROR ON TAPE9 IN SUBROUTINE TRIALS	ABORT	A system I/O failure was encountered on scratch disk (TAPE9) used to store available planes and objective function values as a function of stage. Rerun job or consult systems analyst.

TABLE A-7 (Cont'd)

Message	Severity	Interpretation
TRIAL CARD IGNORED -- INCORRECT FORMAT	INFO	The key on one of the TRIAL cards is mis-punched. Columns 1-5 should contain "TRIAL".
TRIAL CARD IGNORED -- TOO MANY STAGES REQUESTED	INFO	The number of stages requested for a trial war exceeds the number of stages specified in the original run of ATACM1.
TRIAL CARD IGNORED -- TOO MANY PLANES OF ONE TYPE SPECIFIED	INFO	The initial number of planes specifed on a TRIAL card for one of the aircraft types exceeds the maximum grid level specified for that aircraft type in the original run of ATACM1.
RUN ABORTED DUE TO FATAL ERRORS	ABORT	Indicates previously diagnosed ERRORS necessitate job abortion.

ATACM2

As depicted in the logical flowchart of Figure A-7, the required inputs to ATACM2 are a TRIAL-TAPE written by a previous run of ATACM1 and the TRIAL cards requesting war evaluations. In addition to the outcomes of the requested trial wars, ATACM2 can print the inputs, the strategies, and the states used in the original run of ATACM1 to generate the TRIAL-TAPE being read.

INPUTS

The TRIAL-TAPE is the primary input to ATACM2 and is assigned the logical file name "TAPE4". The input deck to ATACM2 consists of three card types described previously under the discussion of ATACM1's inputs. The first card in the input deck must be a RUN card, the last card must be a FINIS card, and the remaining cards must be TRIAL cards. The formats of all three cards are identical to those shown in Figure A-3 with the exception that RUN card parameters in columns 14-80 are ignored by ATACM2. Figure A-8 presents a sample input deck for ATACM2.

OUTPUTS

The outputs produced by ATACM2 are a subset of those produced by ATACM1. By specifying the print parameters described in Table A-3, the user can elect to print any or all of the following outputs under control of the RUN card:

- the input parameters to the original run of ATACM1 which generated the TRIAL-TAPE
- lists of the pure strategies available to Blue and Red in the original run of ATACM1
- a list of the possible states generated in the original run of ATACM1

In addition, for each trial war evaluation requested, ATACM2 produces outputs identical to those described under the outputs of ATACM1 which are controlled by the print options of Table A-4 and displayed in the sample output of Figure A-6.

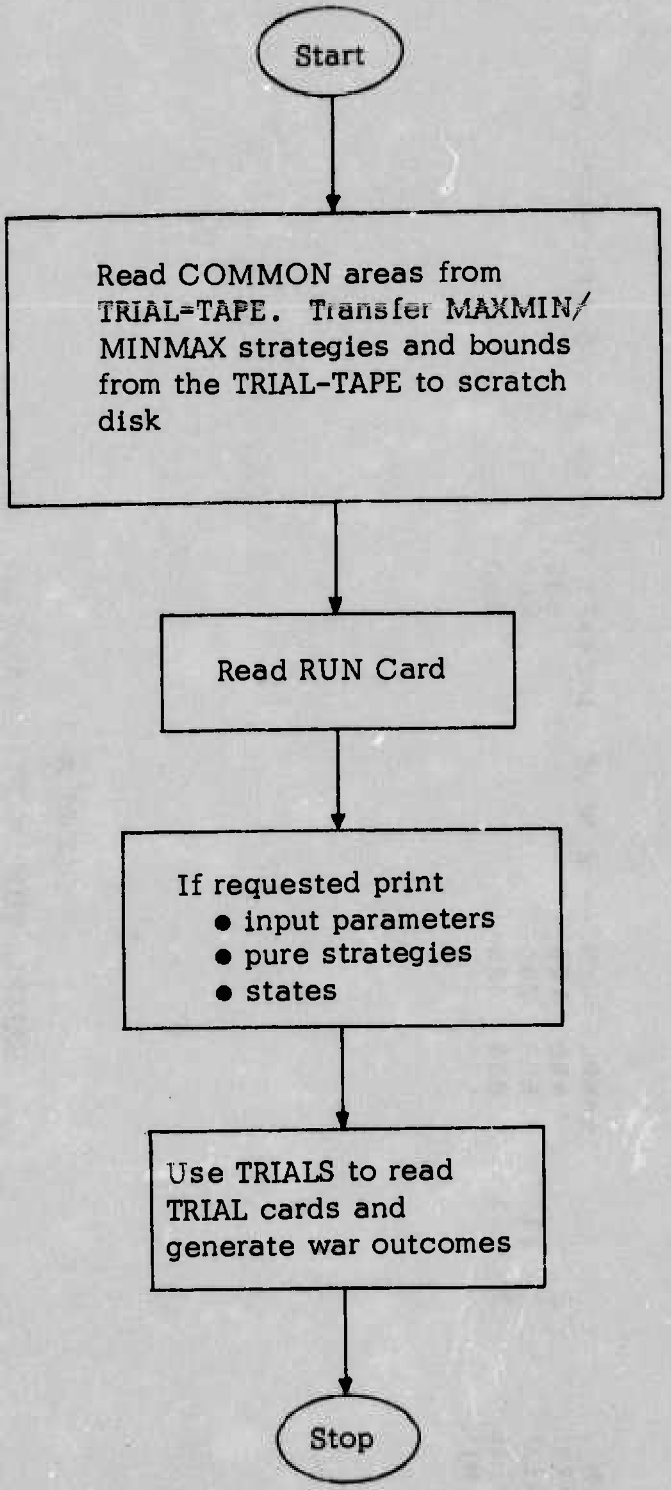

Start
Read COMMON areas from TRIAL=TAPE. Transfer MAXMIN/MINMAX strategies and bounds from the TRIAL-TAPE to scratch disk
Read RUN Card
If requested print
● input parameters
● pure strategies
● states
Use TRIALS to read TRIAL cards and generate war outcomes
Stop

FIGURE A-7

LOGICAL FLOWCHART OF ATACM2

```
RUN                 SAMPLE RUN -- 2 BLUE AIRCRAFT TYPES, 1 RED AIRCRAFT TYPE
TRIAL      2          400  100                    500
TRIAL      1  11      600  200                    600
TRIAL      2    1     800  300                    800
FINIS
```

FIGURE A-8

SAMPLE RUN DECK FOR ATACM2

EXECUTION TIME

The execution time required for ATACM2 is insignificant relative to that required for ATACM1. A typical run requesting the evaluation of 10 trial wars of 10 stages each will generally take less than one minute.

DIAGNOSTIC MESSAGES

Diagnostic messages generated by ATACM2 are classified according to the same scheme described under ATACM1. Table A-8 lists and interprets those messages applicable to ATACM2.

TABLE A-8

DIAGNOSTIC MESSAGES GENERATED BY ATACM2

Message	Severity	Interpretation
RUN CARD MUST PRECEDE TRIAL CARDS	ABORT	The first card in the input deck was not a RUN card.
BUFFER IN ERROR FROM TAPE4 IN MAIN PROGRAM	ABORT	Parity error or system read failure was encountered while reading the TRIAL-TAPE (TAPE4). Rerun job or consult systems analyst.
BUFFER OUT ERROR TO TAPE7 IN MAIN PROGRAM	ABORT	System write failure was encountered while writing MAXMIN/MINMAX objective function values on scratch disk (TAPE7). Rerun job or consult systems analyst.
BUFFER IN ERROR FROM TAPE7 IN SUBROUTINE TRIALS	ABORT	Parity error or system read failure was encountered while reading MAXMIN/MINMAX objective function values from scratch disk (TAPE7). Rerun job or consult systems analyst.
BUFFER OUT ERROR TO TAPE8 IN MAIN PROGRAM	ABORT	System write failure was encountered while writing MAXMIN/MINMAX plays on scratch disk (TAPE8). Rerun job or consult systems analyst.
BUFFER IN ERROR FROM TAPE8 IN MAIN PROGRAM	ABORT	Parity error or system read failure was encountered while reading MAXMIN/MINMAX plays on scratch disk (TAPE8). Rerun job or consult systems analyst.

TABLE A-8 (Cont'd)

Message	Severity	Interpretation
DISK I/O ERROR ON TAPE9 IN SUBROUTINE TRIALS	ABORT	A system I/O failure was encountered on scratch disk (TAPE9) used to store available planes and objective function values as a function of stage. Rerun job or consult systems analyst.
TRIAL CARD IGNORED -- INCORRECT FORMAT	INFO	The key on one of the TRIAL cards is mis-punched. Columns 1-5 should contain "TRIAL".
TRIAL CARD IGNORED -- TOO MANY STAGES REQUESTED	INFO	The number of stages requested for a trial war exceeds the number of stages specified in the original run of ATACM1.
TRIAL CARD IGNORED -- TOO MANY PLANES OF ONE TYPE SPECIFIED	INFO	The initial number of planes specified on a TRIAL card for one of the aircraft types exceeds the maximum grid level specified for that aircraft type in the original run of ATACM1.

APPENDIX B

PROGRAMMING DOCUMENTATION

This appendix presents programming documentation for ATACM1 and ATACM2. Following sections describe the storage requirements for the two programs and potential problems associated with converting them to a computer system different from the current CDC6600. In addition, FORTRAN listings and definitions of the most frequently used variable names are provided.

STORAGE REQUIREMENTS

In its current form ATACM1 requires approximately 54,000 words of core storage, of which about 14,000 are used for program instructions and 40,000 are used for array storage. In addition, approximately 2,000,000 words of scratch disk storage, as described in Table B-1, are required for a representative run of the model.

ATACM2 requires approximately 40,000 words of core storage, of which about 8,000 are used for program instructions and 32,000 are used for array storage. Scratch disk requirements for ATACM2 include those shown for files TAPE7, TAPE8, and TAPE9 in Table B-1 -- approximately 16,000 words for a representative run.

CONVERSION TO A DIFFERENT COMPUTER

Both ATACM1 and ATACM2 are coded in CDC 6600 FORTRAN EXTENDED which is generally compatible with FORTRAN compilers available on other major computer systems. Use of those capabilities of the CDC FORTRAN which are unique or less standard was purposely avoided to minimize the problem of program conversions. Subroutine and variable names are limited to six characters, the standard H specification is used in FORMAT statements for the output of Hollerith strings, multiple assignment statements are not used, etc. ENCODE and DECODE statements are used extensively in the subroutine READIN but, if required, they could be eliminated by imposing more strict rules upon the order of the cards in the input deck.

Assuming the problem of FORTRAN compatibility can be resolved, the only remaining obstacles to conversion are the core and disk storage

TABLE B-1

SCRATCH DISK STORAGE REQUIREMENTS FOR ATACM1

Logical File Name	Access Mode	Number of Records	Number of Words per Record	Total Number of Words Required**
TAPE1	Sequential	NSTATE*	2·NRST·NRST	2,000,000
TAPE2	Random	NSTAGE	2·NSTATE	8,000
TAPE3	Random	NSTAGE	2·NSTATE	8,000
TAPE7	Sequential	NSTAGE	2·NSTATE	8,000
TAPE8	Sequential	NSTAGE	2·NSTATE	8,000
TAPE9	Sequential	NSTAGE	10	100

*Mnemonic variables are defined below:

NSTAGE	=	number of stages
NSTATE	=	number of states
NBST	=	number of Blue strategies
NRST	=	number of Red strategies

**Representative value computed with:

NSTAGE	=	10
NSTATE	=	400
NBST	=	50
NRST	=	50

requirements described above. Unfortunately, these requirements will probably increase rather than decrease in a conversion to another machine because of the large 60-bit word used in the CDC6600. The number of words required for program instructions after conversion to a 32 or 36-bit word machine (e.g. IBM or UNIVAC) could be as many as twice (60/32 or 60/36) the number currently required. Most array and disk storage requirements would be unaffected by a smaller word, the one notable exception being the storage used for battle assessments. In the current versions of ATACM1 and ATACM2 ten integer values are required to characterize the results of each one-stage battle assessment and these are packed into two words. Eight unsigned integer values are stored in one word (7 bits each), and the remaining two signed values are stored in the other word (30 bits each). The most natural allocation of these values on machines with 32 or 36-bit words would require four words -- two for the eight values and one word each for the other two. In a representative run such as that described in Table B-1, the one-stage battle assessments for a single state which currently occupy 5,000 words would require 10,000 words in the converted programs. Analogously, the total amount of scratch disk required to store the one-stage battle assessments for all states on TAPE1 (see Table B-1) would increase from 2 to 4 million words. To illustrate the impact of smaller words, Table B-2 summarizes current and estimated storage requirements for both ATACM1 and ATACM2 before and after conversion.

One final consideration in the conversion problem is the possibility of reducing the size or complexity of the air campaign which can be simulated in order to fit the model to the storage available. The major array used for variable storage in both ATACM1 and ATACM2 is a singly dimensioned vector called XARRAY. The location of values in XARRAY are assigned dynamically depending upon the number of strategies, number of missions, and number of states addressed in the scenario being simulated. The total number of words in XARRAY which are required for a particular run is generated by

$$\text{Total Words Required in XARRAY} = \text{NBST}\cdot\text{NBM}+\text{NRST}\cdot\text{NRM}+2\cdot\text{NBST}\cdot\text{NRST}+10\cdot\text{NSTATE} \qquad \text{(B-1)}$$

where

NBST =	number of Blue strategies
NRST =	number of Red strategies
NBM =	number of Blue missions per strategy
NRM =	number of Red missions per strategy
NSTATE =	number of states

TABLE B-2

CURRENT VS. ESTIMATED STORAGE
REQUIREMENTS AFTER CONVERSION TO A 32-BIT WORD COMPUTER*

	ATACM1		ATACM2	
	Before	After	Before	After
Core Storage				
Instructions	14K	28K	8K	16K
Data Arrays	40K	45K	32K	37K
Total	54K	73K	40K	53K
Scratch Disk				
TAPE1	2,000K	4,000K	**	**
TAPE2	8K	8K	**	**
TAPE3	8K	8K	**	**
TAPE7	8K	8K	8K	8K
TAPE8	8K	8K	8K	8K
TAPE9	.1K	.1K	.1K	.1K
Total	2,032K	4,032K	16K	16K

*Based upon the same representative run used in Table B-1.

**Not used by ATACM2.

In the current versions XARRAY is dimensioned to be 25,000 words long, the value of NWORK. If calculations using Equation B-1 indicate fewer than 25,000 words are adequate for the type of runs which will be made on a different computer, both NWORK and the size of XARRAY can be reduced to a more compatible value*. The result would be a commensurate reduction in the data array storage requirements shown in Table B-2. Although additional reductions beyond those possible by changing NWORK can be achieved by reducing other array dimensions, such changes require a more detailed understanding of the program structure and would yield considerably smaller savings relative to the effort required.

FORTRAN LISTINGS

Figures B-1 and B-2 present FORTRAN listings of ATACM1 and ATACM2 as written for the CDC6600 system. Comment cards in the listings describe the functions of the various subroutine while Table B-3 lists and defines the variable names used most frequently in the two programs.

*The only restriction is that NWORK can not be reduced below 6600 -- the length of NALOCS in READIN.

FIGURE B-1

ATACM1 LISTING

```
      PROGRAM ATACM1 (OUTPUT,TAPE1=65,TAPE2,TAPE3,TAPE4=65,
     1TAPE5,TAPE6=OUTPUT,TAPE7=65,TAPE8=65,TAPE9=65,TAPE10=65)
C
C
C                            ATACM1
C
C        VERSION 1.0                        MAY, 1975
C
C
C         /ATACM1/ IS THE MAIN PROGRAM OF THE ACDA TACTICAL AIR
C         CAMPAIGN MODEL WRITTEN BY KETRON, INC. FOR THE US ARMS
C         CONTROL AND DISARMAMENT AGENCY.  /ATACM1/ CALLS THE
C         SUBROUTINES USED TO PRODUCE OPTIMAL CONSERVATIVE PLAYS
C         AND ASSOCIATED MAXMIN/MINMAX OBJECTIVE FUNCTION VALUES
C         FOR ALL STAGES AND STATES OF THE CAMPAIGN.
C
      COMMON /INPUT/ IMISS(8,4,2),IGRID(11,4,2),LASTP,NALOC(8,4),
     1NFRAC(4,2),NSHL(2),NSTAGE,NDAPST,CASF(4,2),IPRINT(8),
     1ITITLE(6),VALU(4,2),PKBD(4,4,2),PKBDES(4,4,2),XGRID(11,4,2),
     1PKAD(4,4,2),PKADES(4,4,2),PKBA(4,4,2),PKAA(4,4,2),
     1PKESBD(4,4,2),PKESAD(4,4,2),PKSH(4,2),PKNS(4,2),REIN(4,2,100),
     1WGHT(100,2),XSORT(8,4,2),NDIV(2),OWGHT(5),NRSAM(2),NFSAM(2),
     1PKRS(4,2),PKFS(4,2),ABAF(8,4,2),DIVFP(2),PKBAFS(4,2),
     1PKAAFS(4,2),PKAARS(4,2),PKAEFS(4,2),PKAERS(4,2),PKBEFS(4,2),
     1PKFAFS(4,2),PKRAFS(4,2),PKRARS(4,2),DFPRED(4,2),FEBA(2,28),
     1REINF(4,2,100)
      COMMON /WORKN/ NTYPE(2),NMISS(4,2),NMISST(2),NGRID(4,2),
     1IBLURD(2),NSTRAT(4,2),NFULST(2),NSTAT,NINGAM,IMPNT(32,2),
     1ITPNT(32,2),IDEM(8),NSTRTC(2),IRA(100),JRA(100),LUNI,LUNO,
     1NWORK,NSTAT2,ISINT(500,2),IDINT(100,2),INPNT(32,2)
      COMMON /WORKL/ LOCST(500,2),LOCBG,LOCEG,KOCBG,KOCEG,LOCBVR,
     1LOCBVB,LOCEVR,LOCEVB,LOCBPR,LOCBPB,LOCEPR,LOCEPB,
     1KOCBVB,KOCEVB,KOCBVR,KOCEVR,JOCBVB,JOCEVB,JOCBVR,JOCEVR,
     1IOCBVB,IOCEVB,IOCBVR,IOCEVR
      COMMON /WORK/ XARRAY(25000)
      COMMON /ERROR/ IERR
      DIMENSION INPUTZ(2680),WORKNZ(1640),WORKLZ(1024),WORKZ(25000)
      DIMENSION IARRAY(1)
      DIMENSION NAMES(6)
      EQUIVALENCE (IMISS,INPUTZ),(NTYPE,WORKNZ),(LOCST,WORKLZ)
      EQUIVALENCE (XARRAY,IARRAY,WORKZ)
      DATA((NAMES(I),I=1,5)=5HSTART,6HREADIN,6HPRNTIN,6HBATTLE,5HGAMES)
      DATA(NAMES(6)=6HTRIALS)
      DATA(IERR=0)
      CALL SECOND(T)
      WRITE(LUNO,100) T,NAMES(1)
  100 FORMAT(/1X,F9.3,24H CPU SECONDS USED AFTER ,A10)
      T=TIOCALL(X)
      WRITE(LUNO,110) T,NAMES(1)
  110 FORMAT(1X,F9.3,24H I/O SECONDS USED AFTER ,A10/)
      IERR=0
      CALL READIN
      CALL SECOND(T)
      WRITE(LUNO,100) T,NAMES(2)
      T=TIOCALL(X)
      WRITE(LUNO,110) T,NAMES(2)
      IF(IERR .EQ. 1) CALL ERR(1)
      IF(IPRINT(1) .EQ. 0) CALL PRNTIN
      CALL SECOND(T)
```

FIGURE B-1 (cont'd)

```
      WRITE(LUNO,100) T,NAMES(3)
      T=TIOCALL(X)
      WRITE(LUNO,110) T,NAMES(3)
      CALL INIT
      CALL SECOND(T)
      WRITE(LUNO,100) T,NAMES(4)
      T=TIOCALL(X)
      WRITE(LUNO,110) T,NAMES(4)
      IF(IERR .EQ. 1) CALL ERR(1)
      CALL GAMES
      CALL SECOND(T)
      WRITE(LUNO,100) T,NAMES(5)
      T=TIOCALL(X)
      WRITE(LUNO,110) T,NAMES(5)
      IF(IERR .EQ. 1) CALL ERR(1)
C
C         WRITE COMMON BLOCKS TO THE TRIAL-TAPE TO BE USED FOR
C         SUBSEQUENT SENSITIVITY ANALYSES
C
      BUFFER OUT (4,1) (INPUTZ,INPUTZ(2680))
      IF(UNIT(4)) 200,150,150
150   CALL ERR(11)
200   BUFFER OUT (4,1) (WORKNZ,WORKNZ(1640))
      IF(UNIT(4)) 300,150,150
300   BUFFER OUT (4,1) (WORKLZ,WORKLZ(1024))
      IF(UNIT(4)) 400,150,150
400   BUFFER OUT (4,1) (WORKZ,WORKZ(NWORK))
      IF(UNIT(4)) 500,150,150
C
C         WRITE VALUE AND PLAY OUTCOMES FOR EACH STAGE AND
C         STATE ON THE TRIAL-TAPE AND ON SCRATCH DISK
C
500   DO 600 I=1,NSTAGE
      CALL READMS(2,IARRAY(LOCBVB),NSTAT2,I)
      BUFFER OUT (4,1) (IARRAY(LOCBVB),IARRAY(LOCEVR))
      IF(UNIT(4)) 520,150,150
520   CALL READMS(3,IARRAY(LOCBPB),NSTAT2,I)
      BUFFER OUT (4,1) (IARRAY(LOCBPB),IARRAY(LOCEPR))
      IF(UNIT(4)) 530,150,150
530   BUFFER OUT (7,1) (IARRAY(LOCBVB),IARRAY(LOCEVR))
      IF(UNIT(7)) 550,540,540
540   CALL ERR(12)
550   BUFFER OUT (8,1) (IARRAY(LOCBPB),IARRAY(LOCEPR))
      IF(UNIT(8)) 600,560,560
560   CALL ERR(13)
600   CONTINUE
      CALL TRIALS
      CALL SECOND(T)
      WRITE(LUNO,100) T,NAMES(6)
      T=TIOCALL(X)
      WRITE(LUNO,110) T,NAMES(6)
      END
```

FIGURE B-1 (cont'd)

```
      SUBROUTINE BATTLE
C
C         /BATTLE/ COMPUTES THE RESULTS OF A ONE-STAGE ENGAGEMENT
C         BETWEEN SPECIFIED NUMBERS OF BLUE AND RED GROUND AND AIR
C         FORCES ALLOCATED TO MISSIONS ACCORDING TO SPECIFIED
C         STRATEGIES.
C
      COMMON /INPUT/ IMISS(8,4,2),IGRID(11,4,2),LASTP,NALOC(8,4),
     1NFRAC(4,2),NSHL(2),NSTAGE,NDAPST,CASF(4,2),IPRINT(8),
     1ITITLE(6),VALU(4,2),PKBD(4,4,2),PKBDES(4,4,2),XGRID(11,4,2),
     1PKAD(4,4,2),PKADES(4,4,2),PKHA(4,4,2),PKAA(4,4,2),
     1PKESBD(4,4,2),PKESAD(4,4,2),PKSH(4,2),PKNS(4,2),REIN(4,2,100),
     1WGHT(100,2),XSORT(8,4,2),NDIV(2),OWGHT(5),NRSAM(2),NFSAM(2),
     1PKRS(4,2),PKFS(4,2),ABAF(8,4,2),DIVFP(2),PKBAFS(4,2),
     1PKAAFS(4,2),PKAARS(4,2),PKAEFS(4,2),PKAERS(4,2),PKBEFS(4,2),
     1PKFAFS(4,2),PKRAFS(4,2),PKRARS(4,2),DFPRED(4,2),FEBA(2,28),
     1REINF(4,2,100)
      COMMON /WORKN/ NTYPE(2),NMISS(4,2),NMISST(2),NGRID(4,2),
     1IBLURD(2),NSTRAT(4,2),NFULST(2),NSTAT,NINGAM,IMPNT(32,2),
     1ITPNT(32,2),IDEM(8),NSTRTC(2),IRA(100),JRA(100),LUNI,LUNO,
     1NWORK,NSTAT2,ISINT(500,2),IDINT(100,2),INPNT(32,2)
      COMMON /WORKL/ LOCST(500,2),LOCBG,LOCEG,KOCBG,KOCEG,LOCBVR,
     1LOCBVB,LOCEVR,LOCEVB,LOCBPR,LOCBPB,LOCEPR,LOCEPB,
     1KOCBVB,KOCEVB,KOCBVR,KOCEVR,JOCBVB,JOCEVB,JOCBVR,JOCEVR,
     1IOCBVB,IOCEVB,IOCBVR,IOCEVR
C
C
C              CODES FOR AIR MISSIONS
C
C              1 - CAS
C              2 - ABA
C              3 - BD
C              4 - ABD
C              5 - CAS ESCORT
C              6 - ABA ESCORT
C              7 - FORWARD DEFENSE SUPPRESSION
C              8 - REAR DEFENSE SUPPRESSION
C              9 - NOTHING
C
C
      COMMON /WORK/ XARRAY(25000)
      DIMENSION IARRAY(1)
      EQUIVALENCE (XARRAY,IARRAY)
      COMMON /BPARM/ CNP(4,2),IBR(2),XNP(9,4,2),OBJEC(2,5)
      DIMENSION TNP(9,2),REMP(4,2),TCASO(2),TOTFP(2),RFSAM(2),RRSAM(2)
      DIMENSION YNP(9,4,2),TFIRE(2),CASO(2),CNPV(4)
      DATA(XNP=72(0.)),(YNP=72(0.))
      TMOVE=0.
      DO 50 K=1,2
      TCASO(K)=0.
      TOTFP(K)=0.
      RFSAM(K)=NFSAM(K)
      RRSAM(K)=NRSAM(K)
   50 CONTINUE
      DO 900 N=1,NDAPST
      XMOVE=0.
      DO 100 M=1,9
      DO 100 K=1,2
      TNP(M,K)=0.
```

FIGURE B-1 (cont'd)

```
  100   CONTINUE
C
C           DETERMINE NUMBER OF SORTIES ALLOCATED TO EACH MISSION
C
        DO 110 K=1,2
        CASO(K)=0.
        TFIRE(K)=NDIV(K)*DIVFP(K)
        IUP=NMISST(K)
        IST=IBR(K)
        IST=LOCST(IST,K)-1
        DO 110 I=1,IUP
        IST=IST+1
        L=INPNT(I,K)
        M=IMPNT(I,K)
        J=ITPNT(I,K)
        XNP(M,J,K)=XARRAY(IST)*CNP(J,K)*XSORT(L,J,K)
        YNP(M,J,K)=XNP(M,J,K)
        TNP(M,K)=TNP(M,K)+XNP(M,J,K)
  110   CONTINUE
C
C
C           BATTLE ASSESSMENT
C
C
        DO 200 K=1,2
        L=3-K
        XFSAM=RFSAM(L)
        XRSAM=RRSAM(L)
        IHI1=NTYPE(K)
C
C           FORWARD SAM SUPPRESSORS VS FORWARD SAMS
C
        IF(XFSAM .EQ. 0.) GO TO 125
        SUMS=0.
        DO 115 I=1,IHI1
        SUMS=SUMS+XNP(7,I,K)*PKFS(I,L)
  115   CONTINUE
        RFSAM(L)=XFSAM*EXP(-SUMS/XFSAM)
C
C           FORWARD SAMS VS FORWARD SAM SUPPRESSORS
C
        XOPP =TNP(7,K)
        IF(XOPP .EQ. 0.) GO TO 125
        XNENG=AMIN1(XOPP,XFSAM)
        RO=XNENG/XOPP
        DO 120 I=1,IHI1
        XNPS=XNP(7,I,K)
        XN=RO*XNPS*PKFAFS(I,K)
        XNP(7,I,K)=XNP(7,I,K)-XN
  120   CONTINUE
C
C           REMAINING FORWARD SAMS VS REAR SAM SUPPRESSORS
C
  125   XOPP=TNP(8,K)
        IF(XOPP .EQ. 0.) GO TO 200
        XFSAM=RFSAM(L)
        IF(XFSAM .EQ. 0.) GO TO 140
        XNENG=AMIN1(XOPP,XFSAM)
        RO=XNENG/XOPP
```

FIGURE B-1 (cont'd)

```
      DO 130 I=1,IHI1
      XNPS=XNP(8,I,K)
      XN=RO*XNPS*PKRAFS(I,K)
      XNP(8,I,K)=XNPS-XN
      XOPP=XOPP-XN
  130 CONTINUE
C
C          REMAINING REAR SAM SUPPRESSORS VS REAR SAMS
C
      IF(XOPP .LE. 0.) GO TO 200
  140 IF(XRSAM .EQ. 0.) GO TO 155
      SUMS=0.
      DO 145 I=1,IHI1
      SUMS=SUMS+XNP(8,I,K)*PKRS(I,L)
  145 CONTINUE
      RRSAM(L)=XRSAM*EXP(-SUMS/XRSAM)
C
C          REAR SAMS VS REMAINING REAR SAM SUPPRESSORS
C
      XNENG=AMIN1(XOPP,XRSAM)
      RO=XNENG/XOPP
      DO 150 I=1,IHI1
      XNPS=XNP(8,I,K)
      XN=RO*XNPS*PKRARS(I,K)
      XNP(8,I,K)=XNPS-XN
      XOPP=XOPP-XN
  150 CONTINUE
C
C          FORWARD SAMS VS RETURNING REAR SAM SUPPRESSORS
C
      IF(XOPP .LE. 0.) GO TO 200
  155 IF(XFSAM .EQ. 0.) GO TO 200
      XNENG=AMIN1(XOPP,XFSAM)
      RO=XNENG/XOPP
      DO 160 I=1,IHI1
      XNPS=XNP(8,I,K)
      XN=RO*XNPS*PKRAFS(I,K)
      XNP(8,I,K)=XNPS-XN
  160 CONTINUE
  200 CONTINUE
C
      DO 700 K=1,2
      L=3-K
      IHI1=NTYPE(K)
      IHI2=NTYPE(L)
      XFSAM=RFSAM(L)
      XRSAM=RRSAM(L)
C
C          FORWARD SAMS VS CAS ESCORTS
C
      RO1=1.
      XATT=TNP(5,K)
      XOPP=TNP(3,L)
      IF(XATT .EQ. 0.) GO TO 310
      IF(XFSAM .EQ. 0.) GO TO 250
      XNENG=AMIN1(XATT,XFSAM)
      RO=XNENG/XATT
      DO 225 I=1,IHI1
      XNPS=XNP(5,I,K)
```

FIGURE B-1 (cont'd)

```
      XN=RO*XNPS*PKBEFS(I,K)
      XNP(5,I,K)=XNPS-XN
      XATT=XATT-XN
225   CONTINUE
C
C         CAS ESCORTS VS BD
C
      IF(XATT .LE. 0.) GO TO 310
250   IF(XOPP .EQ. 0.) GO TO 310
      XNENG=AMIN1(XATT,XOPP)
      RO=XNENG/XOPP
      ROA=RO/XATT
      RO1=1.-RO
      DO 300 I=1,IHI1
      DO 300 J=1,IHI2
      XNPSB=XNP(5,I,K)
      XNPSR=XNP(3,J,L)
      XN=XNPSB*XNPSR*ROA
      XNP(5,I,K)=XNPSB-PKESBD(J,I,K)*XN
      XNP(3,J,L)=XNPSR-PKBDES(I,J,L)*XN
300   CONTINUE
310   DO 350 J=1,IHI2
      REMP(J,L)=YNP(3,J,L)*RO1
350   CONTINUE
C
C         FORWARD SAMS VS CAS
C
      XATT=XOPP*RO1
      XOPP=TNP(1,K)
      RO1=1.
      IF(XOPP .EQ. 0.) GO TO 410
      IF(XFSAM .EQ. 0.) GO TO 370
      XNENG=AMIN1(XOPP,XFSAM)
      RO=XNENG/XOPP
      DO 360 I=1,IHI1
      XNPS=XNP(1,I,K)
      XN=RO*XNPS*PKBAFS(I,K)
      XNP(1,I,K)=XNPS-XN
      YNP(1,I,K)=XNP(1,I,K)
      XOPP=XOPP-XN
360   CONTINUE
C
C         BD NOT ENGAGED VS CAS
C
      IF(XOPP .LE. 0.) GO TO 410
370   IF(XATT .EQ. 0.) GO TO 410
      XNENG=AMIN1(XATT,XOPP)
      RO=XNENG/XOPP
      ROA=RO/XATT
      RO1=1.-RO
      DO 400 I=1,IHI1
      DO 400 J=1,IHI2
      XNPS=XNP(1,I,K)
      XN=REMP(J,L)*XNPS*ROA
      XNP(3,J,L)=XNP(3,J,L)-PKBD(I,J,L)*XN
      XNP(1,I,K)=XNPS-PKBA(J,I,K)*XN
400   CONTINUE
C
C         ACCUMULATE CAS ORDNANCE DELIVERED BY CAS NOT ENGAGED
```

FIGURE B-1 (cont'd)

```
C
  410 DO 420 I=1,IHI1
      XNPC=YNP(1,I,K)*RO1
      CASO(K)=CASO(K)+XNPC*CASF(I,K)
      TFIRE(L)=TFIRE(L)-XNPC*DFPRED(I,K)
  420 CONTINUE
C
C         FORWARD SAMS VS ABA ESCORTS
C
      RO1=1.
      XATT=TNP(6,K)
      XOPP=TNP(4,L)
      IF(XATT .EQ. 0.) GO TO 540
      IF(XFSAM .EQ. 0.) GO TO 470
      XNENG=AMIN1(XATT,XFSAM)
      RO=XNENG/XATT
      DO 460 I=1,IHI1
      XNPS=XNP(6,I,K)
      XN=RO*XNPS*PKAEFS(I,K)
      XNP(6,I,K)=XNPS-XN
      XATT=XATT-XN
  460 CONTINUE
C
C         REAR SAMS VS ABA ESCORTS
C
  470 IF(XATT .LE. 0.) GO TO 540
      IF(XRSAM .EQ. 0.) GO TO 490
      XNENG=AMIN1(XATT,XRSAM)
      RO=XNENG/XATT
      DO 480 I=1,IHI1
      XNPS=XNP(6,I,K)
      XN=RO*XNPS*PKAERS(I,K)
      XNP(6,I,K)=XNPS-XN
      XATT=XATT-XN
  480 CONTINUE
C
C         ABA ESCORTS VS ABD
C
      IF(XATT .LE. 0.) GO TO 540
  490 IF(XOPP .EQ. 0.) GO TO 510
      XNENG=AMIN1(XATT,XOPP)
      RO=XNENG/XOPP
      ROA=RO/XATT
      RO1=1.-RO
      DO 500 I=1,IHI1
      DO 500 J=1,IHI2
      XNPSB=XNP(6,I,K)
      XNPSR=XNP(4,J,L)
      XN=XNPSB*XNPSR*ROA
      XNN=PKESAD(J,I,K)*XN
      XNP(6,I,K)=XNPSB-XNN
      XATT=XATT-XNN
      XNP(4,J,L)=XNPSR-PKADES(I,J,L)*XN
  500 CONTINUE
C
C         FORWARD SAMS VS RETURNING ABA ESCORTS
C
  510 IF(XATT .LE. 0.) GO TO 540
      IF(XFSAM .EQ. 0.) GO TO 540
```

FIGURE B-1 (cont'd)

```
      XNENG=AMIN1(XATT,XFSAM)
      RO=XNENG/XATT
      DO 520 I=1,IHI1
      XNPS=XNP(6,I,K)
      XNP(6,I,K)=XNPS-XNPS*RO*PKAEFS(I,K)
  520 CONTINUE
  540 DO 550 J=1,IHI2
      REMP(J,L)=YNP(4,J,L)*RO1
  550 CONTINUE
C
C         FORWARD SAMS VS ABA
C
      XATT=XOPP*RO1
      XOPP=TNP(2,K)
      RO1=1.
      IF(XOPP .EQ. 0.) GO TO 640
      IF(XFSAM .EQ. 0.) GO TO 570
      XNENG=AMIN1(XOPP,XFSAM)
      RO=XNENG/XOPP
      DO 560 I=1,IHI1
      XNPS=XNP(2,I,K)
      XN=RO*XNPS*PKAAFS(I,K)
      XNP(2,I,K)=XNPS-XN
      YNP(2,I,K)=XNP(2,I,K)
      XOPP=XOPP-XN
  560 CONTINUE
C
C         REAR SAMS VS ABA
C
      IF(XOPP .LE. 0.) GO TO 640
  570 IF(XRSAM .EQ. 0.) GO TO 590
      XNENG=AMIN1(XOPP,XRSAM)
      RO=XNENG/XOPP
      DO 580 I=1,IHI1
      XNPS=XNP(2,I,K)
      XN=RO*XNPS*PKAARS(I,K)
      XNP(2,I,K)=XNPS-XN
      YNP(2,I,K)=XNP(2,I,K)
      XOPP=XOPP-XN
  580 CONTINUE
C
C         ABD NOT ENGAGED VS ABA
C
      IF(XOPP .LE. 0.) GO TO 640
  590 IF(XATT .EQ. 0.) GO TO 610
      XNENG=AMIN1(XATT,XOPP)
      RO=XNENG/XOPP
      ROA=RO/XATT
      RO1=1.-RO
      DO 600 I=1,IHI1
      DO 600 J=1,IHI2
      XNPS=XNP(2,I,K)
      XN=REMP(J,L)*XNPS*ROA
      XNN=PKAA(J,I,K)*XN
      XNP(2,I,K)=XNPS-XNN
      XNP(4,J,L)=XNP(4,J,L)-PKAD(I,J,L)*XN
      XOPP=XOPP-XNN
  600 CONTINUE
C
```

FIGURE B-1 (cont'd)

```
C          FORWARD SAMS VS RETURNING ABA
C
  610 IF(XOPP .LE. 0.) GO TO 640
      IF(XFSAM .LE. 0.) GO TO 640
      XNENG=AMIN1(XOPP,XFSAM)
      RO=XNENG/XOPP
      DO 620 I=1,IHI1
      XNPS=XNP(2,I,K)
      XNP(2,I,K)=XNPS-XNPS*RO*PKAAFS(I,K)
  620 CONTINUE
  640 DO 650 I=1,IHI1
      REMP(I,K)=YNP(2,I,K)*RO1
  650 CONTINUE
  700 CONTINUE
C
C          COMPUTE AIRCRAFT ON GROUND KILLED BY ABA NOT ENGAGED
C
      DO 800 K=1,2
      L=3-K
      IHI1=NTYPE(K)
      IHI2=NTYPE(L)
      ATTK=0.
      TARG=0.
      SUMS=0.
      SUMN=0.
      DO 715 J=1,IHI2
      IUP=NMISS(J,L)
      CNP(J,L)=0.
      CNPV(J)=0.
      DO 710 M=1,IUP
      IM=IMISS(M,J,L)
      CP=XNP(IM,J,L)/XSORT(M,J,L)
      CNP(J,L)=CNP(J,L)+CP
      CNPV(J)=CNPV(J)+CP*ABAF(M,J,L)
  710 CONTINUE
      TARG=TARG+CNPV(J)
  715 CONTINUE
      IF(TARG .EQ. 0.) GO TO 800
      NTARG=TARG
      TARGS=AMINO(NSHL(L),NTARG)
      TARGN=NTARG-TARGS
      RS=TARGS/TARG
      RN=1.-RS
      DO 720 I=1,IHI1
      CP=REMP(I,K)
      ATTK=ATTK+CP
      SUMS=SUMS+CP*RS*PKSH(I,L)
      SUMN=SUMN+CP*RN*PKNS(I,L)
  720 CONTINUE
      IF(ATTK .EQ. 0.) GO TO 800
      TSNK=0.
      IF(TARGS .EQ. 0.) GO TO 730
      TSNK=TARGS*EXP(-SUMS/TARGS)
  730 TNNK=0.
      IF(TARGN .EQ. 0.) GO TO 740
      TNNK=TARGN*EXP(-SUMN/TARGN)
  740 FR=1.-(TSNK+TNNK)/TARG
      DO 750 J=1,IHI2
      CNP(J,L)=CNP(J,L)-CNPV(J)*FR
  750 CONTINUE
```

FIGURE B-1 (cont'd)

```
  800 CONTINUE
      DO 805 K=1,2
      TCASO(K)=TCASO(K)+CASO(K)
  805 CONTINUE
      BTFP=AMAX1(0.,TFIRE(1))
      RTFP=AMAX1(0.,TFIRE(2))
      TOTFP(1)=TOTFP(1)+BTFP*CASO(1)
      TOTFP(2)=TOTFP(2)+RTFP*CASO(2)
      IF(FEBA(1,1) .EQ. -1.) GO TO 900
      FRATIO=100.
      IF(RTFP .EQ. 0.) GO TO 810
      FRATIO=BTFP/RTFP
      IF(FRATIO .GT. 100.) FRATIO=100.
      GO TO 815
  810 IF(BTFP .EQ. 0.) FRATIO=1.0
  815 DO 820 I=2,28
      IF(FRATIO .LE. FEBA(1,I)) GO TO 830
  820 CONTINUE
      GO TO 850
  830 XMOVE=FEBA(2,I-1)+(FRATIO-FEBA(1,I-1))*(FEBA(2,I)-FEBA(2,I-1))/
     1(FEBA(1,I)-FEBA(1,I-1))
  850 TMOVE=TMOVE+XMOVE
  900 CONTINUE
C
C-----ASSIGN OBJECTIVE FUNCTION VALUES
C
      DO 920 K=1,2
      OBJEC(K,1)=TCASO(K)
      OBJEC(K,2)=TOTFP(K)
  920 CONTINUE
      OBJEC(1,3)=TMOVE/2.
      OBJEC(2,3)=-OBJEC(1,3)
      RETURN
      END


      SUBROUTINE BETAS
C
C         /BETAS/ COMPUTES THE WEIGHTS USED FOR LINEAR INTERPOLATION
C         BETWEEN GRID POINTS IN THE STATE SPACE.
C
      COMMON /INPUT/ IMISS(8,4,2),IGRID(11,4,2),LASTP,NALOC(8,4),
     1NFRAC(4,2),NSHL(2),NSTAGE,NDAPST,CASF(4,2),IPRINT(8),
     1ITITLE(6),VALU(4,2),PKBD(4,4,2),PKBDES(4,4,2),XGRID(11,4,2),
     1PKAD(4,4,2),PKADES(4,4,2),PKBA(4,4,2),PKAA(4,4,2),
     1PKESBD(4,4,2),PKESAD(4,4,2),PKSH(4,2),PKNS(4,2),REIN(4,2,100),
     1WGHT(100,2),XSORT(8,4,2),NDIV(2),OWGHT(5),NRSAM(2),NFSAM(2),
     1PKRS(4,2),PKFS(4,2),ABAF(8,4,2),DIVFP(2),PKBAFS(4,2),
     1PKAAFS(4,2),PKAARS(4,2),PKAEFS(4,2),PKAERS(4,2),PKBEFS(4,2),
     1PKFAFS(4,2),PKRAFS(4,2),PKRARS(4,2),DFPRED(4,2),FEBA(2,28),
     1REINF(4,2,100)
      COMMON /WORKN/ NTYPE(2),NMISS(4,2),NMISST(2),NGRID(4,2),
     1IBLURD(2),NSTRAT(4,2),NFULST(2),NSTAT,NINGAM,IMPNT(32,2),
     1ITPNT(32,2),IDEM(8),NSTRTC(2),IRA(100),JRA(100),LUNI,LUNO,
     1NWORK,NSTAT2,ISINT(500,2),IDINT(100,2),INPNT(32,2)
```

FIGURE B-1 (cont'd)

```
      COMMON /WORKL/ LOCST(500,2),LOCBG,LOCEG,KOCBG,KOCEG,LOCBVR,
     1LOCBVB,LOCEVR,LOCEVB,LOCBPR,LOCBPB,LOCEPR,LOCEPB,
     1KOCBVB,KOCEVB,KOCBVR,KOCEVR,JOCBVB,JOCEVB,JOCBVR,JOCEVR,
     1IOCBVB,IOCEVB,IOCBVR,IOCEVR
      COMMON /BPARM/ CNP(8),IB,IR,XNP(9,4,2),OBJEC(2,5)
      COMMON /INTERP/ BETA(2,8),IBETA(2,8),NDEX(8),IHI(8)
      DIMENSION YGRID(11,8)
      EQUIVALENCE (XGRID,YGRID)
      DO 200 K=1,8
      DO 100 M=2,11
      IF(CNP(K)-YGRID(M,K)) 150,150,100
  100 CONTINUE
      M=11
  150 N=M-1
      IBETA(1,K)=N-1
      IBETA(2,K)=M-1
      DIF=YGRID(M,K)-YGRID(N,K)
      IF(DIF .EQ. 0.) GO TO 175
      BETA(1,K)=(YGRID(M,K)-CNP(K))/DIF
      GO TO 190
  175 BETA(1,K)=1.
      IBETA(2,K)=0
  190 BETA(2,K)=1.-BETA(1,K)
  200 CONTINUE
      IF(IPRINT(8) .EQ. 0) RETURN
      WRITE(LUNO,900) IBETA
  900 FORMAT(/,8H IBETAS=,2(8I5/8X))
      WRITE(LUNO,910) BETA
  910 FORMAT(7H BETAS=,2(8F5.2/7X))
      RETURN
      END
```

```
      SUBROUTINE ERR(K)
C
C          /ERR/ IS CALLED TO PRINT A DIAGNOSTIC OR ERROR MESSAGE.
C
      COMMON /INPUT/ IMISS(8,4,2),IGRID(11,4,2),LASTP,NALOC(8,4),
     1NFRAC(4,2),NSHL(2),NSTAGE,NDAPST,CASF(4,2),IPRINT(8),
     1ITITLE(6),VALU(4,2),PKBD(4,4,2),PKBDES(4,4,2),XGRID(11,4,2),
     1PKAD(4,4,2),PKADES(4,4,2),PKBA(4,4,2),PKAA(4,4,2),
     1PKESBD(4,4,2),PKESAD(4,4,2),PKSH(4,2),PKNS(4,2),REIN(4,2,100),
     1WGHT(100,2),XSORT(8,4,2),NDIV(2),OWGHT(5),NRSAM(2),NFSAM(2),
     1PKRS(4,2),PKFS(4,2),ABAF(8,4,2),DIVFP(2),PKBAFS(4,2),
     1PKAAFS(4,2),PKAARS(4,2),PKAEFS(4,2),PKAERS(4,2),PKBEFS(4,2),
     1PKFAFS(4,2),PKRAFS(4,2),PKRARS(4,2),DFPRED(4,2),FEBA(2,28),
     1REINF(4,2,100)
      COMMON /WORKN/ NTYPE(2),NMISS(4,2),NMISST(2),NGRID(4,2),
     1IBLURD(2),NSTRAT(4,2),NFULST(2),NSTAT,NINGAM,IMPNT(32,2),
     1ITPNT(32,2),IDEM(8),NSTRTC(2),IRA(100),JRA(100),LUNI,LUNO,
     1NWORK,NSTAT2,ISINT(500,2),IDINT(100,2),INPNT(32,2)
      COMMON /WORKL/ LOCST(500,2),LOCBG,LOCEG,KOCBG,KOCEG,LOCBVR,
     1LOCBVB,LOCEVR,LOCEVB,LOCBPR,LOCBPB,LOCEPR,LOCEPB,
     1KOCBVB,KOCEVB,KOCBVR,KOCEVR,JOCBVB,JOCEVB,JOCBVR,JOCEVR,
     1IOCBVB,IOCEVB,IOCBVR,IOCEVR
      COMMON /ERROR/ IERR
```

FIGURE B-1 (cont'd)

```
      DIMENSION IMESSG(6,30),IERROR(3),IETYPE(30)
      DATA(IMESSG=180(1H ))
      DATA(IERROR=10H***INFO***,10H***ERROR**,10H***ABORT**)
      DATA(IMESSG(1,1)=31HRUN ABORTED DUE TO FATAL ERRORS)
      DATA(IMESSG(1,2)=28HDATA CARD KEY NOT RECOGNIZED)
      DATA(IMESSG(1,3)=45HTOO MANY PURE STRATEGIES TO BE STORED IN CORE)
      DATA(IMESSG(1,4)=41HTOO MANY STATES TO STORE VALUES AND PLAYS)
      DATA(IMESSG(1,5)=43HNOT ENOUGH COMMON AREA TO STORE GAME MATRIX)
      DATA(IMESSG(1,6)=44HBUFFER OUT ERROR TO TAPE1 IN SUBROUTINE INIT)
      DATA(IMESSG(1,7)=40HUSER REQUESTED ONLY A DATA CHECK AND TIM)
      DATA(IMESSG(5,7)=10HE ESTIMATE)
      DATA(IMESSG(1,8)=40HSUM OF SPECIFIED ALLOCATIONS EXCEEDS 1.0)
      DATA(IMESSG(1,9)=32HSMALLEST GRID LEVEL MUST BE ZERO)
      DATA(IMESSG(1,10)=40HBUFFER IN ERROR FROM TAPE1 IN SUBROUTINE)
      DATA(IMESSG(5,10)=6H GAMES)
      DATA(IMESSG(1,11)=41HBUFFER OUT ERROR TO TAPE4 IN MAIN PROGRAM)
      DATA(IMESSG(1,12)=41HBUFFER OUT ERROR TO TAPE7 IN MAIN PROGRAM)
      DATA(IMESSG(1,13)=41HBUFFER OUT ERROR TO TAPE8 IN MAIN PROGRAM)
      DATA(IMESSG(1,14)=40HBUFFER IN ERROR FROM TAPE7 IN SUBROUTINE)
      DATA(IMESSG(5,14)=7H TRIALS)
      DATA(IMESSG(1,15)=40HBUFFER IN ERROR FROM TAPE8 IN SUBROUTINE)
      DATA(IMESSG(5,15)=7H TRIALS)
      DATA(IMESSG(1,16)=33HRUN CARD MUST PRECEDE TRIAL CARDS)
      DATA(IMESSG(1,17)=38HTRIAL CARD IGNORED -- INCORRECT FORMAT)
      DATA(IMESSG(1,18)=40HTRIAL CARD IGNORED -- TOO MANY STAGES RE)
      DATA(IMESSG(5,18)=7HQUESTED)
      DATA(IMESSG(1,19)=40HTRIAL CARD IGNORED -- TOO MANY PLANES OF)
      DATA(IMESSG(5,19)=19H ONE TYPE SPECIFIED)
      DATA(IMESSG(1,20)=40HDISK I/O ERROR ON TAPE9 IN SUBROUTINE TR)
      DATA(IMESSG(5,20)=4HIALS)
      DATA(IMESSG(1,21)=40HBUFFER OUT ERROR ON TAPE9 IN SUBROUTINE )
      DATA(IMESSG(5,21)=6HREADIN)
      DATA(IMESSG(1,22)=42HBUFFER IN ERROR FROM TAPE4 IN MAIN PROGRAM)
      DATA(IMESSG(1,23)=40HIF ALL ALLOCATIONS ARE SPECIFIED THEY MU)
      DATA(IMESSG(5,23)=13HST SUM TO 1.0)
      DATA(IMESSG(1,24)=38HI/O ERROR ON TAPE10 IN SUBROUTINE INIT)
      DATA(IMESSG(1,25)=40HSTRATEGIES NOT SPECIFIED THRU THE LAST S)
      DATA(IMESSG(5,25)=16HTAGE OF CAMPAIGN)
      DATA(IMESSG(1,26)=40HBUFFER IN ERROR ON TAPE9 IN SUBROUTINE I)
      DATA(IMESSG(5,26)=3HNIT)
      DATA(IETYPE=3,2,6(3),2,7(3),3(1),4(3),7(3))
      IDERR=IETYPE(K)
      WRITE(LUNO,100) IERROR(IDERR),(IMESSG(I,K),I=1,6)
  100 FORMAT(/,1X,A10,1X,6A10,/)
      IF(IDERR .EQ. 3) STOP
      IF(IDERR .EQ. 1) RETURN
      IERR=1
      RETURN
      END
```

```
      SUBROUTINE GAMES
C
C           /GAMES/ PERFORMS A DYNAMIC PROGRAMMING BACKWARD PASS
C           COMPUTING FOR EACH STAGE AND STATE THE MAXMIN AND MINMAX
C           STRATEGIES AND ASSOCIATED OBJECTIVE FUNCTION VALUES FOR
C           BOTH BLUE AND RED.
```

FIGURE B-1 (cont'd)

```
C
      COMMON /INPUT/ IMISS(8,4,2),IGRID(11,4,2),LASTP,NALOC(8,4),
     1NFRAC(4,2),NSHL(2),NSTAGE,NDAPST,CASF(4,2),IPRINT(8),
     1ITITLE(6),VALU(4,2),PKBD(4,4,2),PKBDES(4,4,2),XGRID(11,4,2),
     1PKAD(4,4,2),PKADES(4,4,2),PKBA(4,4,2),PKAA(4,4,2),
     1PKESBD(4,4,2),PKESAD(4,4,2),PKSH(4,2),PKNS(4,2),REIN(4,2,100),
     1WGHT(100,2),XSORT(8,4,2),NDIV(2),OWGHT(5),NRSAM(2),NFSAM(2),
     1PKRS(4,2),PKFS(4,2),ABAF(8,4,2),DIVFP(2),PKBAFS(4,2),
     1PKAAFS(4,2),PKAARS(4,2),PKAEFS(4,2),PKAERS(4,2),PKBEFS(4,2),
     1PKFAFS(4,2),PKRAFS(4,2),PKRARS(4,2),DFPRED(4,2),FEBA(2,28),
     1REINF(4,2,100)
      COMMON /WORKN/ NTYPE(2),NMISS(4,2),NMISST(2),NGRID(4,2),
     1IBLURD(2),NSTRAT(4,2),NFULST(2),NSTAT,NINGAM,IMPNT(32,2),
     1ITPNT(32,2),IDEM(8),NSTRTC(2),IRA(100),JRA(100),LUNI,LUNO,
     1NWORK,NSTAT2,ISINT(500,2),IDINT(100,2),INPNT(32,2)
      COMMON /WORKL/ LOCST(500,2),LOCBG,LOCEG,KOCBG,KOCEG,LOCBVR,
     1LOCBVB,LOCEVR,LOCEVB,LOCBPR,LOCBPB,LOCEPR,LOCEPB,
     1KOCBVB,KOCEVB,KOCBVR,KOCEVR,JOCBVB,JOCEVB,JOCBVR,JOCEVR,
     1IOCBVB,IOCEVB,IOCBVR,IOCEVR
      COMMON /INTERP/ BETA(2,8),IBETA(2,8),J1,J2,J3,J4,J5,J6,J7,J8,
     1JHI1,JHI2,JHI3,JHI4,JHI5,JHI6,JHI7,JHI8
      COMMON /IPARM/ DELTA(8),JBETA(2,8,128),XBETA(2,8,128),IBIT(8)
      COMMON /ROUND/ JDEX(4,2,2)
      DIMENSION MVEC(8),NVEC(8)
      EQUIVALENCE(MVEC,JDEX(1,1,1)),(NVEC,JDEX(1,1,2))
      COMMON /WORK/ XARRAY(25000)
      DIMENSION JHI(8),JVEC(8),NDEX(8),BMIN(500),RMAX(500),IARRAY(1)
      DIMENSION IVERT(8,256),IBSINT(500),IRSINT(500)
      EQUIVALENCE (ISINT(1,1),IBSINT),(ISINT(1,2),IRSINT)
      EQUIVALENCE(XARRAY,IARRAY),(NDEX,J1),(JHI,JHI1)
      DIMENSION SEIN(8,100),SEINF(8,100)
      EQUIVALENCE (REIN,SEIN),(REINF,SEINF)
      COMMON /TEMP/ IPPNT(8),TEIN(8),TEINF(8),NPTT
      DATA(JDEX=16(0)),(JVEC=8(0)),(JHI=8(1)),(IVERT=2048(1))
      NPTT=NTYPE(1)+NTYPE(2)
      MPTT=NPTT
      M=4-NTYPE(1)
      DO 25 NN=1,NPTT
      JHI(NN)=2
      N=NN
      IF(N .GT. NTYPE(1)) N=N+M
      IPPNT(NN)=N
   25 CONTINUE
      IF(IPPNT(NPTT) .EQ. 8) MPTT=MPTT-1
      ICNT=0
      DO 50 J8=1,JHI8
      DO 50 J7=1,JHI7
      DO 50 J6=1,JHI6
      DO 50 J5=1,JHI5
      DO 50 J4=1,JHI4
      DO 50 J3=1,JHI3
      DO 50 J2=1,JHI2
      DO 50 J1=1,JHI1
      ICNT=ICNT+1
      DO 50 NN=1,NPTT
      IVERT(NN,ICNT)=NDEX(NN)
   50 CONTINUE
      LPTT=ICNT
      IHI1=NFULST(2)
```

FIGURE B-1 (cont'd)

```
      IHI2=IHI1-1
      IHI3=NFULST(1)
      LVB=LOCBVB-1
      LVR=LOCBVR-1
      LPB=LOCBPB-1
      LPR=LOCBPR-1
      KVB=KOCBVB-1
      KVR=KOCBVR-1
      JVB=JOCBVB-1
      JVR=JOCBVR-1
      IVB=IOCBVB-1
      IVR=IOCBVR-1
      DO 900 I=1,NSTAGE
      MSLOC=NSTAGE-I+1
      NSLOC=MSLOC+1
      IDINTB=IDINT(MSLOC,1)
      IDINTR=IDINT(MSLOC,2)
      WGHTB=WGHT(MSLOC,1)
      WGHTR=WGHT(MSLOC,2)
      DO 75 NN=1,NPTT
      N=IPPNT(NN)
      TEINF(N)=SEINF(N,NSLOC)
      TEIN(N)=SEIN(N,NSLOC)/DELTA(N)+1.5
 75   CONTINUE
      REWIND 1
      DO 800 J=1,NSTAT
      IF(IPRINT(7) .NE. 0) WRITE(LUNO,80) MSLOC,J
 80   FORMAT(/,7H STAGE=,I3,3X,6HSTATE=,I3)
      BUFFER IN (1,1) (IARRAY(LOCBG),IARRAY(KOCEG))
      IF(UNIT(1)) 200,100,100
 100  CALL ERR(10)
C
C         COMPUTE INTERPOLATED VALUES AND PLAYS FOR EACH STATE
C
 200  DO 210 K=1,IHI3
      BMIN(K)=1.E10
 210  CONTINUE
      DO 220 K=1,IHI1
      RMAX(K)=-1.E10
 220  CONTINUE
      ICNT=0
      DO 400 K=LOCBG,LOCEG,IHI1
      ICNT=ICNT+1
      IF(IBSINT(ICNT) .EQ. IDINTB) GO TO 300
      BMIN(ICNT)=-1.E10
      GO TO 400
 300  IHI=K+IHI2
      JCNT=0
      DO 390 L=K,IHI
      JCNT=JCNT+1
      IF(IRSINT(JCNT) .EQ. IDINTR) GO TO 305
      RMAX(JCNT)=1.E10
      GO TO 390
 305  IWORD=IARRAY(L)
      JWORD=IARRAY(L+NINGAM)
      DO 310 NN=1,NPTT
      N=IPPNT(NN)
      LEVEL=LBYT(IBIT(N),7,JWORD)
```

FIGURE B-1 (cont'd)

```
      LEVEL=TEINF(N)*LEVEL+TEIN(N)
      IF(LEVEL .LT. 1) LEVEL=1
      IF(LEVEL .GT. 128) LEVEL=128
      DO 310 M=1,2
      IBETA(M,N)=JBETA(M,N,LEVEL)
      BETA(M,N)=XBETA(M,N,LEVEL)
  310 CONTINUE
      IOBJ=LBYT(31,29,IWORD)
      JOBJ=LBYT(1,29,IWORD)
      IF(LBYT(60,1,IWORD) .EQ. 1) IOBJ=-IOBJ
      IF(LBYT(30,1,IWORD) .EQ. 1) JOBJ=-JOBJ
      CHECKB=IOBJ*WGHTB-JOBJ*WGHTR
      CHECKR=CHECKB
      DO 340 LL=1,LPTT
      ALPHA=1.
      DO 320 NN=1,NPTT
      N=IPPNT(NN)
      M=IVERT(NN,LL)
      JVEC(N)=IBETA(M,N)
      ALPHA=ALPHA*BETA(M,N)
  320 CONTINUE
      IF(ALPHA .EQ. 0.) GO TO 340
      ISTAT=JVEC(8)+1
      DO 330 NN=1,MPTT
      N=IPPNT(NN)
      ISTAT=ISTAT+JVEC(N)*IDEM(N)
  330 CONTINUE
      CHECKB=CHECKB+ALPHA*XARRAY(LVB+ISTAT)
      CHECKR=CHECKR+ALPHA*XARRAY(LVR+ISTAT)
  340 CONTINUE
      IF(CHECKB .GE. BMIN(ICNT)) GO TO 350
      BMIN(ICNT)=CHECKB
  350 IF(CHECKR .LE. RMAX(JCNT)) GO TO 390
      RMAX(JCNT)=CHECKR
  390 CONTINUE
  400 CONTINUE
C
C         STORE BLUES MAXMIN PLAY
C
      XMAX=-1.E10
      DO 410 N=1,IHI3
      IF(XMAX .GE. BMIN(N)) GO TO 410
      XMAX=BMIN(N)
      IBPLAY=N
  410 CONTINUE
      XARRAY(KVB+J)=XMAX
      IARRAY(LPB+J)=IBPLAY
C
C         STORE REDS MINMAX PLAY
C
      XMIN=1.E10
      DO 420 N=1,IHI1
      IF(XMIN .LE. RMAX(N)) GO TO 420
      XMIN=RMAX(N)
      IRPLAY=N
  420 CONTINUE
      XARRAY(KVR+J)=XMIN
      IARRAY(LPR+J)=IRPLAY
C
```

FIGURE B-1 (cont'd)

```
C          COMPUTE AND STORE MAXMIN FOR BLUE
C
      ILO=LOCBG+IHI1*(IBPLAY-1)
      IHI=ILO+IHI1-1
      JCNT=0
      XMIN=1.E10
      DO 600 K=ILO,IHI
      JCNT=JCNT+1
      IF(IRSINT(JCNT) .NE. IDINTR) GO TO 600
      IWORD=IARRAY(K)
      JWORD=IARRAY(K+NINGAM)
      IOBJ=LBYT(31,29,IWORD)
      JOBJ=LBYT(1,29,IWORD)
      IF(LBYT(60,1,IWORD) .EQ. 1) IOBJ=-IOBJ
      IF(LBYT(30,1,IWORD) .EQ. 1) JOBJ=-JOBJ
      DO 510 NN=1,NPTT
      N=IPPNT(NN)
      M=(N-1)/4+1
      LEVEL=LBYT(IBIT(N),7,JWORD)
      LEVEL=TEINF(N)*LEVEL+TEIN(N)
      IF(LEVEL .LT. 1) LEVEL=1
      IF(LEVEL .GT. 128) LEVEL=128
      MVEC(N)=JBETA(M,N,LEVEL)
  510 CONTINUE
      IS=ISTATE(MVEC)
      CHECK=IOBJ*WGHTB-JOBJ*WGHTR+XARRAY(JVB+IS)
      IF(CHECK .GE. XMIN) GO TO 600
      XMIN=CHECK
  600 CONTINUE
      XARRAY(IVB+J)=XMIN
      IF(IPRINT(7) .NE. 0) WRITE(LUNO,605) XMIN,IBPLAY
  605 FORMAT(8H MAXMIN=,F10.0,2X,5HPLAY=,I3)
C
C          COMPUTE AND STORE MINMAX FOR RED
C
      ILO=LOCBG+IRPLAY-1
      IHI=ILO+(IHI3-1)*IHI1
      ICNT=0
      XMAX=-1.E10
      DO 700 K=ILO,IHI,IHI1
      ICNT=ICNT+1
      IF(IBSINT(ICNT) .NE. IDINTB) GO TO 700
      IWORD=IARRAY(K)
      JWORD=IARRAY(K+NINGAM)
      IOBJ=LBYT(31,29,IWORD)
      JOBJ=LBYT(1,29,IWORD)
      IF(LBYT(60,1,IWORD) .EQ. 1) IOBJ=-IOBJ
      IF(LBYT(30,1,IWORD) .EQ. 1) JOBJ=-JOBJ
      DO 610 NN=1,NPTT
      N=IPPNT(NN)
      MM=2-(N-1)/4
      LEVEL=LBYT(IBIT(N),7,JWORD)
      LEVEL=TEINF(N)*LEVEL+TEIN(N)
      IF(LEVEL .LT. 1) LEVEL=1
      IF(LEVEL .GT. 128) LEVEL=128
      NVEC(N)=JBETA(MM,N,LEVEL)
  610 CONTINUE
      IS=ISTATE(NVEC)
      CHECK=IOBJ*WGHTB-JOBJ*WGHTR+XARRAY(JVR+IS)
```

FIGURE B-1 (cont'd)

```
      IF(CHECK .LE. XMAX) GO TO 700
      XMAX=CHECK
  700 CONTINUE
      XARRAY(IVR+J)=XMAX
      IF(IPRINT(7) .NE. 0) WRITE(LUNO,705) XMAX,IRPLAY
  705 FORMAT(8H MINMAX=,F10.0,2X,5HPLAY=,I3)
  800 CONTINUE
C
C         WRITE PLAYS AND VALUES ON RA MASS STORAGE
C
      CALL WRITMS(3,IARRAY(LOCBPB),NSTAT2,MSLOC)
      CALL WRITMS(2,IARRAY(IOCBVB),NSTAT2,MSLOC)
      DO 850 J=1,NSTAT2
      IARRAY(LVB+J)=IARRAY(KVB+J)
      IARRAY(JVB+J)=IARRAY(IVB+J)
  850 CONTINUE
  900 CONTINUE
      RETURN
      END


      SUBROUTINE INIT
C
C          /INIT/ ASSIGNS COUNTERS AND POINTERS, COMPUTES THE PURE
C          STRATEGIES AND THE NUMBER OF STATES, AND COMPUTES AND STORES
C          ON MASS STORAGE THE BATTLE ASSESSMENTS FOR EACH STATE AND
C          PURE STRATEGY COMBINATION.
C
      COMMON /INPUT/ IMISS(8,4,2),IGRID(11,4,2),LASTP,NALOC(8,4),
     1NFRAC(4,2),NSHL(2),NSTAGE,NDAPST,CASF(4,2),IPRINT(8),
     1ITITLE(6),VALU(4,2),PKBD(4,4,2),PKBDES(4,4,2),XGRID(11,4,2),
     1PKAD(4,4,2),PKADES(4,4,2),PKBA(4,4,2),PKAA(4,4,2),
     1PKESBD(4,4,2),PKESAD(4,4,2),PKSH(4,2),PKNS(4,2),REIN(4,2,100),
     1WGHT(100,2),XSORT(8,4,2),NDIV(2),OWGHT(S),NRSAM(2),NFSAM(2),
     1PKRS(4,2),PKFS(4,2),ABAF(8,4,2),DIVFP(2),PKBAFS(4,2),
     1PKAAFS(4,2),PKAARS(4,2),PKAEFS(4,2),PKAERS(4,2),PKBEFS(4,2),
     1PKFAFS(4,2),PKRAFS(4,2),PKRARS(4,2),DFPRED(4,2),FEBA(2,28),
     1REINF(4,2,100)
      COMMON /WORKN/ NTYPE(2),NMISS(4,2),NMISST(2),NGRID(4,2),
     1IBLURD(2),NSTRAT(4,2),NFULST(2),NSTAT,NINGAM,IMPNT(32,2),
     1ITPNT(32,2),IDEM(8),NSTRTC(2),IRA(100),JRA(100),LUNI,LUNO,
     1NWORK,NSTAT2,ISINT(500,2),IDINT(100,2),INPNT(32,2)
      COMMON /WORKL/ LOCST(500,2),LOCBG,LOCEG,KOCBG,KOCEG,LOCBVR,
     1LOCBVB,LOCEVR,LOCEVB,LOCBPR,LOCBPB,LOCEPR,LOCEPB,
     1KOCBVB,KOCEVB,KOCBVR,KOCEVR,JOCBVB,JOCEVB,JOCBVR,JOCEVR,
     1IOCBVB,IOCEVB,IOCBVR,IOCEVR
      COMMON /WORK/ XARRAY(25000)
      DIMENSION IARRAY(1)
      DIMENSION KEVEL(8)
      EQUIVALENCE (XARRAY,IARRAY)
      DIMENSION STRATS(8,200,4),JGRID(11,8),YGRID(11,8)
      EQUIVALENCE (XARRAY(18601),STRATS)
      EQUIVALENCE (IGRID,JGRID),(XGRID,YGRID)
      COMMON /SPARM/ MFRAC,MMISS,IBR,ITYPE,NSTOR(8),IPNT(8)
      COMMON /BPARM/ CNP(4,2),IB,IR,XNP(9,4,2),OBJEC(2,5)
      COMMON /IPARM/ DELTA(4,2),JBETA(2,8,128),XBETA(2,8,128),IBIT(8)
```

FIGURE B-1 (cont'd)

```
      COMMON /INTERP/ BETA(2,8),IBETA(2,8),NDEX(8),IHI(8)
      COMMON /SINTVL/ IRLO(500),IRHI(500)
      DIMENSION ISTOR(4)
      DATA(NWORK=25000)
      DATA(IBIT=50,43,36,29,22,15,8,1)
      DATA(NSTRAT=8(0)),(NMISS=8(0)),(XNP=72(0.))
C
      DO 100 I=1,NWORK
      IARRAY(I)=0
  100 CONTINUE
      DO 110 K=1,2
      DO 110 J=1,4
      DO 110 I=1,11
      XGRID(I,J,K)=IGRID(I,J,K)
  110 CONTINUE
C
C         ASSIGN COUNTERS AND POINTERS
C
      DO 140 K=1,2
      L=0
      DO 135 J=1,4
      DO 115 I=1,8
      IF(IMISS(I,J,K) .EQ. 0) GO TO 120
      L=L+1
      INPNT(L,K)=I
      IMPNT(L,K)=IMISS(I,J,K)
      ITPNT(L,K)=J
  115 CONTINUE
  120 NMISS(J,K)=I-1
      DO 125 I=2,11
      IF(IGRID(I,J,K) .EQ. 0) GO TO 130
  125 CONTINUE
  130 NGRID(J,K)=I-1
      DELTA(J,K)=XGRID(I-1,J,K)/127.
  135 CONTINUE
      NMISST(K)=L
  140 CONTINUE
C
C         COMPUTE INTERPOLATION PARAMETERS
C
      DO 160 I=1,128
      L=I-1
      DO 150 K=1,2
      DO 150 J=1,4
      CNP(J,K)=L*DELTA(J,K)
  150 CONTINUE
      CALL BETAS
      DO 160 K=1,2
      DO 160 J=1,8
      JBETA(K,J,I)=IBETA(K,J)
      XBETA(K,J,I)=BETA(K,J)
  160 CONTINUE
      DO 165 K=1,2
      DO 165 J=1,4
      IF(DELTA(J,K) .EQ. 0.) DELTA(J,K)=1.
  165 CONTINUE
      CALL OPENMS(2,IRA,100,0)
      CALL OPENMS(3,JRA,100,0)
C
```

FIGURE B-1 (cont'd)

```
C          GENERATE PURE STRATEGIES FOR BLUE AND RED
C
      ILOC=1
      REWIND 9
      DO 400 K=1,2
      IBR=K
      IUP1=NTYPE(K)
      IUP3=NSTRTC(K)
      JCNT=0
      KASTP=1
      DO 360 II=1,IUP3
      BUFFER IN (9,1) (LASTP,NALOC(8,4))
      IF(UNIT(9)) 175,170,170
 170  CALL ERR(26)
 175  DO 180 J=KASTP,LASTP
      IDINT(J,K)=LASTP
 180  CONTINUE
      KASTP=LASTP+1
      DO 300 J=1,IUP1
      ITYPE=J
      MMISS=NMISS(J,K)
      MFRAC=NFRAC(J,K)
      IHI2=MMISS
      ICNT=0
      DO 200 I=1,IHI2
      IPNT(I)=0
      IF(NALOC(I,J) .EQ. -1) GO TO 190
      MMISS=MMISS-1
      MFRAC=MFRAC-NALOC(I,J)
      IF(MFRAC .LT. 0) CALL ERR(8)
      NSTOR(I)=NALOC(I,J)
      GO TO 200
 190  ICNT=ICNT+1
      IPNT(ICNT)=I
 200  CONTINUE
      IF(MMISS .EQ. 0 .AND. MFRAC .GT. 0) CALL ERR(23)
      CALL STRAT
 300  CONTINUE
      IHI1=NSTRAT(1,K)
      IHI2=NSTRAT(2,K)
      IHI3=NSTRAT(3,K)
      IHI4=NSTRAT(4,K)
      IF(IHI1 .EQ. 0) IHI1=1
      IF(IHI2 .EQ. 0) IHI2=1
      IF(IHI3 .EQ. 0) IHI3=1
      IF(IHI4 .EQ. 0) IHI4=1
      DO 360 I=1,IHI1
      DO 360 J=1,IHI2
      DO 360 L=1,IHI3
      DO 360 M=1,IHI4
      JCNT=JCNT+1
      LOCST(JCNT,K)=ILOC
      ISINT(JCNT,K)=LASTP
      ISTOR(1)=I
      ISTOR(2)=J
      ISTOR(3)=L
      ISTOR(4)=M
      DO 360 II=1,IUP1
      IDEX=ISTOR(II)
```

FIGURE B-1 (cont'd)

```
      IUP2=NMISS(1,K)
      DO 360 N=1,IUP2
      XARRAY(ILOC)=STRATS(N,IDEX,I1)
      ILOC=ILOC+1
  360 CONTINUE
      NFULST(K)=JCNT
      IF(JCNT .GT. 500) CALL ERR(3)
      IF(K .EQ. 1 .AND. ILOC .GT. NWORK-6400) CALL ERR(3)
      IF(LASTP .LT. NSTAGE) CALL ERR(25)
  400 CONTINUE
      WRITE(LUNO,410)
  410 FORMAT(1H1)
      WRITE(LUNO,420) (IBLURD(K),NFULST(K),K=1,2)
  420 FORMAT(//11H NUMBER OF ,A4,24H PURE STRATEGIES EQUALS ,I5)
      IF(ILOC .GT. NWORK) CALL ERR(3)
C
C         SET POINTERS FOR EACH BLUE STRATEGY TO INDICATE THE RED
C         STRATEGIES WHICH CAN BE PLAYED AGAINST IT
C
      IHI1=NFULST(1)
      IHI2=NFULST(2)
      ILOB=0
      DO 428 I=1,IHI1
      IF(ISINT(I,1) .EQ. ILOB) GO TO 426
      IRL=0
      IBL=ILOB+1
      IBH=ISINT(I,1)
      DO 424 K=IBL,IBH
      DO 422 J=1,IHI2
      IF(ISINT(J,2) .NE. IDINT(K,2)) GO TO 422
      IF(IRL .EQ. 0) IRL=J
      IRH=J
  422 CONTINUE
  424 CONTINUE
      ILOB=IBH
  426 IRLO(I)=IRL
      IRHI(I)=IRH
  428 CONTINUE
C
C         PRINT STRATEGIES UNLESS SUPPRESSED
C
      IF(IPRINT(2) .NE. 0) GO TO 500
      DO 460 K=1,2
      IHI3=NMISST(K)
      IHI1=NFULST(K)
      DO 460 I=1,IHI1
      ILO2=LOCST(I,K)
      IHI2=ILO2+NMISST(K)-1
      IF((I-1)/50*50 .NE. I-1) GO TO 438
      WRITE(LUNO,430) IBLURD(K),(ITPNT(J,K),IMPNT(J,K),J=1,IHI3)
  430 FORMAT(1H1///,17X,A5,15HPURE STRATEGIES//,6H STRAT,4X,4HLAST,
     112X,18HPLANE TYPE/MISSION/,7H NUMBER,3X,5HSTAGE,5X,
     14(10(I1,1H/,I1,2X)/20X))
      WRITE(LUNO,435)
  435 FORMAT(1H )
  438 WRITE(LUNO,440) I,ISINT(I,K),(XARRAY(J),J=ILO2,IHI2)
  440 FORMAT(1X,I5,5X,I3,4X,4(10F5.2,/,18X))
  460 CONTINUE
C
```

FIGURE B-1 (cont'd)

```
C         COMPUTE TOTAL NUMBER OF STATES
C
  500 NSTAT=1
      ICNT=9
      DO 510 K=1,2
      DO 510 J=1,4
      L=3-K
      M=5-J
      ICNT=ICNT-1
      IDEM(ICNT)=NSTAT
      NSTAT=NSTAT*NGRID(M,L)
  510 CONTINUE
      NSTAT2=2*NSTAT
      WRITE(LUNO,520) NSTAT
  520 FORMAT(1H1,//25H NUMBER OF STATES EQUALS ,I7)
      IF(ILOC+5*NSTAT2 .GT. NWORK) CALL ERR(4)
      NG1=NGRID(1,1)
      NG2=NGRID(2,1)
      NG3=NGRID(3,1)
      NG4=NGRID(4,1)
      NG5=NGRID(1,2)
      NG6=NGRID(2,2)
      NG7=NGRID(3,2)
      NG8=NGRID(4,2)
C
C         PRINT STATES UNLESS SUPPRESSED
C
      IF(IPRINT(3) .NE. 0) GO TO 635
      ICNT=0
      DO 630 I1=1,NG1
      DO 630 I2=1,NG2
      DO 630 I3=1,NG3
      DO 630 I4=1,NG4
      DO 630 I5=1,NG5
      DO 630 I6=1,NG6
      DO 630 I7=1,NG7
      DO 630 I8=1,NG8
      ICNT=ICNT+1
      IF((ICNT-1)/50*50 .EQ. ICNT-1) WRITE(LUNO,610)
  610 FORMAT(1H1,//18X,27HLIST OF ALL POSSIBLE STATES,//6H STATE,
     115X,4HBLUE,27X,3HRED,/7H NUMBER,7X,2(19H1     2     3     4,
     111X)/)
      WRITE(LUNO,620) ICNT,JGRID(I1,1),JGRID(I2,2),JGRID(I3,3),
     1JGRID(I4,4),JGRID(I5,5),JGRID(I6,6),JGRID(I7,7),JGRID(I8,8)
  620 FORMAT(1X,I5,4X,2(4I6,6X))
  630 CONTINUE
  635 CALL TIMER
C
C         TERMINATE EXECUTION IF ABORT OPTION IS SPECIFIED
C
      IF(IPRINT(4) .EQ. 1) CALL ERR(7)
      JLOC1=ILOC
C
C         COMPUTE FINAL PAYOFFS FOR EACH STATE AND STORE IN
C         NEXT-STAGE-ARRAYS
C
      DO 645 I1=1,NG1
      DO 645 I2=1,NG2
      DO 645 I3=1,NG3
```

FIGURE B-1 (cont'd)

```
      DO 645 I4=1,NG4
      DO 645 I5=1,NG5
      DO 645 I6=1,NG6
      DO 645 I7=1,NG7
      DO 645 I8=1,NG8
      XARRAY(ILOC)=YGRID(I1,1)*VALU(1,1)+YGRID(I2,2)*VALU(2,1)+
     1YGRID(I3,3)*VALU(3,1)+YGRID(I4,4)*VALU(4,1)-
     1YGRID(I5,5)*VALU(1,2)-YGRID(I6,6)*VALU(2,2)-
     1YGRID(I7,7)*VALU(3,2)-YGRID(I8,8)*VALU(4,2)
      XARRAY(ILOC)=XARRAY(ILOC)*(OWGHT(1)+OWGHT(2))
      ILOC=ILOC+1
  645 CONTINUE
      JLOC2=ILOC-1
      LOCBVB=JLOC1
      LOCEVB=JLOC2
      DO 648 I=JLOC1,JLOC2
      XARRAY(ILOC)=XARRAY(I)
      ILOC=ILOC+1
  648 CONTINUE
      LOCBVR=JLOC2+1
      LOCEVR=ILOC-1
      KOCBVB=LOCEVR+1
      KOCEVB=LOCEVR+NSTAT
      KOCBVR=KOCEVB+1
      KOCEVR=KOCEVB+NSTAT
      JOCBVB=KOCEVR+1
      JOCEVB=KOCEVR+NSTAT
      JOCBVR=JOCEVB+1
      JOCEVR=JOCEVB+NSTAT
      IOCBVB=JOCEVR+1
      IOCEVB=JOCEVR+NSTAT
      IOCBVR=IOCEVB+1
      IOCEVR=IOCEVB+NSTAT
      LOCBPB=IOCEVR+1
      LOCEPB=IOCEVR+NSTAT
      LOCBPR=LOCEPB+1
      LOCEPR=LOCEPB+NSTAT
      DO 650 I=1,NSTAT2
      J=I-1
      XARRAY(JOCBVB+J)=XARRAY(LOCBVB+J)
  650 CONTINUE
C
C          READ OR COMPUTE BATTLE ASSESSMENTS FOR EACH STATE AND
C          PURE STRATEGY COMBINATION AND STORE ON SCRATCH DISK
C
      IHI1=NFULST(1)
      IHI2=NFULST(2)
      NINGAM=IHI1*IHI2
      LOCBG=LOCEPR+1
      LOCEG=LOCEPR+NINGAM
      KOCBG=LOCEG+1
      KOCEG=LOCEG+NINGAM
      IF(KOCEG .GT. NWORK) CALL ERR(5)
      CALL SECOND(T)
      WRITE(LUNO,651) T
  651 FORMAT(/////1X,F9.3,25H CPU SECONDS USED IN INIT/)
      IF(IPRINT(5) .LT. 2) GO TO 665
C
C          IF AVAILABLE READ BATTLE ASSESSMENTS FROM BATTLE-TAPE
C
```

FIGURE B-1 (cont'd)

```
      DO 660 I=1,NSTAT
      BUFFER IN (I0,1) (IARRAY(LOCBG),IARRAY(KOCEG))
      IF(UNIT(I0)) 655,653,653
  653 CALL ERR(24)
  655 BUFFER OUT (I,1) (IARRAY(LOCBG),IARRAY(KOCEG))
      IF(UNIT(I)) 660,658,658
  658 CALL ERR(6)
  660 CONTINUE
      RETURN
C
C          IF BATTLE-TAPE IS NOT AVAILABLE, COMPUTE ASSESSMENTS
C          USING SUBROUTINE /BATTLE/
C
  665 DO 900 I1=1,NG1
      DO 900 I2=1,NG2
      DO 900 I3=1,NG3
      DO 900 I4=1,NG4
      DO 900 I5=1,NG5
      DO 900 I6=1,NG6
      DO 900 I7=1,NG7
      DO 900 I8=1,NG8
      LOCWD=LOCBG-1
      DO 800 IB=1,IHI1
      DO 800 IR=1,IHI2
      IWORD=0
      JWORD=0
      IF(IR .LT. IRLO(IB) .OR. IR .GT. IRHI(IB)) GO TO 780
      CNP(1,1)=YGRID(I1,1)
      CNP(2,1)=YGRID(I2,2)
      CNP(3,1)=YGRID(I3,3)
      CNP(4,1)=YGRID(I4,4)
      CNP(1,2)=YGRID(I5,5)
      CNP(2,2)=YGRID(I6,6)
      CNP(3,2)=YGRID(I7,7)
      CNP(4,2)=YGRID(I8,8)
      IF(IPRINT(6) .EQ. 0) GO TO 680
      WRITE(LUNO,670) IB,IR
  670 FORMAT(/,4H IB=,I3,2X,3HIR=,I3)
      WRITE(LUNO,675) CNP
  675 FORMAT(/,4H CNP,8F9.2)
  680 CALL BATTLE
      LOCLEV=0
      DO 700 K=1,2
      DO 700 J=1,4
      LEVEL=CNP(J,K)/DELTA(J,K)+.5
      LOCLEV=LOCLEV+1
      KEVEL(LOCLEV)=LEVEL
      CALL SBYT(IBIT(LOCLEV),7,JWORD,LEVEL)
  700 CONTINUE
      BOBJ=0.
      ROBJ=0.
      DO 720 J=1,3
      BOBJ=BOBJ+OBJEC(1,J)*OWGHT(J)
      ROBJ=ROBJ+OBJEC(2,J)*OWGHT(J)
  720 CONTINUE
      IOBJ=BOBJ+SIGN(.5,BOBJ)
      JOBJ=ROBJ+SIGN(.5,ROBJ)
      IF(IPRINT(6) .EQ. 0) GO TO 780
      WRITE(LUNO,775) CNP,IOBJ,JOBJ,KEVEL
```

FIGURE B-1 (cont'd)

```
  775   FORMAT(4H CNP,8F9.2/6H IOBJ=,I10,5X,5HJOBJ=,I10,5X,7HLEVELS=,8I4)
  780   IF(IOBJ .GE. 0) GO TO 785
        IOBJ=-IOBJ
        CALL SBYT(60,1,IWORD,1)
  785   CALL SBYT(31,29,IWORD,IOBJ)
        IF(JOBJ .GE. 0) GO TO 790
        JOBJ=-JOBJ
        CALL SBYT(30,1,IWORD,1)
  790   CALL SBYT(1,29,IWORD,JOBJ)
        LOCWD=LOCWO+I
        IARRAY(LOCWD)=IWORD
        IARRAY(LOCWO+NINGAM)=JWORD
  800   CONTINUE
        BUFFER OUT (1,1) (IARRAY(LOCBG),IARRAY(KOCEG))
        IF(UNIT(1)) 875,850,850
  850   CALL ERR(6)
  875   IF(IPRINT(5) .EQ. 0) GO TO 900
  C
  C          WRITE BATTLE ASSESSMENTS ON BATTLE-TAPE IF REQUESTED
  C
        BUFFER OUT (10,1) (IARRAY(LOCBG),IARRAY(KOCEG))
        IF(UNIT(10)) 900,880,880
  880   CALL ERR(24)
  900   CONTINUE
        RETURN
        END


        FUNCTION ISTATE(IV)
  C
  C          /ISTATE/ COMPUTES THE INDEX OF THE STATE CORRESPONDING
  C          TO A SPECIFIED COMBINATION OF GRID POINTS.
  C
        COMMON /INPUT/ IMISS(8,4,2),IGRID(11,4,2),LASTP,NALOC(8,4),
       1NFRAC(4,2),NSHL(2),NSTAGE,NDAPST,CASF(4,2),IPRINT(8),
       1ITITLE(6),VALU(4,2),PKBO(4,4,2),PKBDES(4,4,2),XGRID(11,4,2),
       1PKAD(4,4,2),PKADES(4,4,2),PKBA(4,4,2),PKAA(4,4,2),
       1PKESBO(4,4,2),PKESAD(4,4,2),PKSH(4,2),PKNS(4,2),REIN(4,2,100),
       1WGHT(100,2),XSORT(8,4,2),NDIV(2),OWGHT(5),NRSAM(2),NFSAM(2),
       1PKRS(4,2),PKFS(4,2),ABAF(8,4,2),DIVFP(2),PKBAFS(4,2),
       1PKAAFS(4,2),PKAARS(4,2),PKAEFS(4,2),PKAERS(4,2),PKBEFS(4,2),
       1PKFAFS(4,2),PKRAFS(4,2),PKRARS(4,2),OFPRED(4,2),FEBA(2,28),
       1REINF(4,2,100)
        COMMON /WORKN/ NTYPE(2),NMISS(4,2),NMISST(2),NGRID(4,2),
       1IBLURO(2),NSTRAT(4,2),NFULST(2),NSTAT,NINGAM,IMPNT(32,2),
       1ITPNT(32,2),IDEM(8),NSTRTC(2),IRA(100),JRA(100),LUNI,LUNO,
       1NWORK,NSTAT2,ISINT(500,2),IDINT(100,2),INPNT(32,2)
        COMMON /WORKL/ LOCST(500,2),LOCBG,LOCEG,KOCBG,KOCEG,LOCBVR,
       1LOCBVB,LOCEVR,LOCEVB,LOCBPR,LOCBPB,LOCEPR,LOCEPB,
       1KOCBVB,KOCEVB,KOCBVR,KOCEVR,JOCBVB,JOCEVB,JOCBVR,JOCEVR,
       1IOCBVB,IOCEVB,IOCBVR,IOCEVR
        DIMENSION IV(1)
        ISTATE=IV(8)+1
        DO 100 I=1,7
        ISTATE=ISTATE+IV(I)*IDEM(I)
  100   CONTINUE
        RETURN
        END
```

FIGURE B-1 (cont'd)

```
      SUBROUTINE PRNTIN
C
C         /PRNTIN/ PRINTS THE INPUT PARAMETERS.
C
      COMMON /INPUT/ IMISS(8,4,2),IGRID(11,4,2),LASTP,NALOC(8,4),
     1NFRAC(4,2),NSHL(2),NSTAGE,NDAPST,CASF(4,2),IPRINT(8),
     1ITITLE(6),VALU(4,2),PKBD(4,4,2),PKBDES(4,4,2),XGRID(11,4,2),
     1PKAD(4,4,2),PKADES(4,4,2),PKBA(4,4,2),PKAA(4,4,2),
     1PKESBD(4,4,2),PKESAD(4,4,2),PKSH(4,2),PKNS(4,2),REIN(4,2,100),
     1WGHT(100,2),XSORT(8,4,2),NDIV(2),OWGHT(5),NRSAM(2),NFSAM(2),
     1PKRS(4,2),PKFS(4,2),ABAF(8,4,2),DIVFP(2),PKWAFS(4,2),
     1PKAAFS(4,2),PKAARS(4,2),PKAEFS(4,2),PKAERS(4,2),PKBEFS(4,2),
     1PKFAFS(4,2),PKRAFS(4,2),PKRARS(4,2),DFPRED(4,2),FEBA(2,28),
     1REINF(4,2,100)
      COMMON /WORKN/ NTYPE(2),NMISS(4,2),NMISST(2),NGRID(4,2),
     1IBLURD(2),NSTRAT(4,2),NFULST(2),NSTAT,NINGAM,IMPNT(32,2),
     1ITPNT(32,2),IDEM(8),NSTRTC(2),IRA(100),JRA(100),LUNI,LUNO,
     1NWORK,NSTAT2,ISINT(500,2),IDINT(100,2),INPNT(32,2)
      COMMON /WORKL/ LOCST(500,2),LOCBG,LOCEG,KOCBG,KOCEG,LOCBVR,
     1LOCBVB,LOCEVR,LOCEVB,LOCBPR,LOCBPB,LOCEPR,LOCEPB,
     1KOCBVB,KOCEVB,KOCBVR,KOCEVR,JOCBVB,JOCEVB,JOCBVR,JOCEVR,
     1IOCBVB,IOCEVB,IOCBVR,IOCEVR
      DIMENSION IHEAD(4,25),IM(4,2),XF(4,2),PKS(4,4,2)
      DATA(IHEAD(1, 1)=40H          CAS ESCORT AGAINST BD            )
      DATA(IHEAD(1, 2)=40H          BD AGAINST CAS ESCORT            )
      DATA(IHEAD(1, 3)=40H             CAS AGAINST BD                )
      DATA(IHEAD(1, 4)=40H             BD AGAINST CAS                )
      DATA(IHEAD(1, 5)=40H          ABA ESCORT AGAINST ABD           )
      DATA(IHEAD(1, 6)=40H          ABD AGAINST ABA ESCORT           )
      DATA(IHEAD(1, 7)=40H             ABA AGAINST ABD               )
      DATA(IHEAD(1, 8)=40H             ABD AGAINST ABA               )
      DATA(IHEAD(1, 9)=40H    ABA AGAINST NON-SHELTERED AIRCRAFT     )
      DATA(IHEAD(1,10)=40H      ABA AGAINST SHELTERED AIRCRAFT       )
      DATA(IHEAD(1,11)=40H    FORWARD SAM SUPPRESSOR AGAINST SAM     )
      DATA(IHEAD(1,12)=40H    SAM AGAINST FORWARD SAM SUPPRESSOR     )
      DATA(IHEAD(1,13)=40H      REAR SAM SUPPRESSOR AGAINST SAM      )
      DATA(IHEAD(1,14)=40H      SAM AGAINST REAR SAM SUPPRESSOR      )
      DATA(IHEAD(1,15)=40H             SAM AGAINST CAS               )
      DATA(IHEAD(1,16)=40H          SAM AGAINST CAS ESCORT           )
      DATA(IHEAD(1,17)=40H             SAM AGAINST ABA               )
      DATA(IHEAD(1,18)=40H          SAM AGAINST ABA ESCORT           )
      WRITE(LUNO,100) (ITITLE(I),I=1,6)
  100 FORMAT(1H1,1X,6A10/)
      WRITE(LUNO,102) NSTAGE
  102 FORMAT(1X,18HNUMBER OF STAGES =,I3)
      WRITE(LUNO,104) NDAPST
  104 FORMAT(1X,28HNUMBER OF CYCLES PER STAGE =,I3)
      WRITE(LUNO,110) (IBLURD(K),NTYPE(K),K=1,2)
  110 FORMAT(1X,10HNUMBER OF ,A4,14H PLANE TYPES =,I2)
      WRITE(LUNO,120) (IBLURD(K),NDIV(K),K=1,2)
  120 FORMAT(1X,10HNUMBER OF ,A4,12H DIVISIONS =,I6)
      WRITE(LUNO,125) (IBLURD(K),DIVFP(K),K=1,2)
  125 FORMAT(1X,14HFIREPOWER PER ,A4,11H DIVISION =,F10.4)
      WRITE(LUNO,130) (IBLURD(K),NSHL(K),K=1,2)
  130 FORMAT(1X,10HNUMBER OF ,A4,11H SHELTERS =,I6)
      WRITE(LUNO,140) (IBLURD(K),NFSAM(K),K=1,2)
  140 FORMAT(1X,10HNUMBER OF ,A4,15H FORWARD SAMS =,I6)
      WRITE(LUNO,150) (IBLURD(K),NRSAM(K),K=1,2)
  150 FORMAT(1X,10HNUMBER OF ,A4,12H REAR SAMS =,I6)
```

FIGURE B-1 (cont'd)

```
      WRITE(LUNO,160) (OWGHT(I),I=1,3)
  160 FORMAT(1X,29HOBJECTIVE FUNCTION WEIGHTS = ,9HCAS ORD -,F6.2,
     13X,9HTOTL FP -,F6.2,3X,6HFEBA -,F6.2)
      WRITE(LUNO,200)
  200 FORMAT(/// ,29X,21HMISSIONS ASSIGNED AND/,28X,
     123HASSOCIATED SORTIE RATES//)
      WRITE(LUNO,210)
  210 FORMAT(15X,15HBLUE PLANE TYPE,21X,14HRED PLANE TYPE/
     12(10X,25H1         2         3         4)/)
      DO 300 I=1,8
      DO 240 K=1,2
      DO 240 J=1,4
      IF(IMISS(I,J,K) .NE. 0) GO TO 250
  240 CONTINUE
      GO TO 310
  250 DO 260 K=1,2
      DO 260 J=1,4
      IM(J,K)=0
      XF(J,K)=0.
      IF(IMISS(I,J,K) .NE. 0) IM(J,K)=IMISS(I,J,K)
      IF(XSORT(I,J,K) .NE. 0.) XF(J,K)=XSORT(I,J,K)
  260 CONTINUE
      WRITE(LUNO,280) ((IM(J,K),XF(J,K),J=1,4),K=1,2)
  280 FORMAT(4X,2(3X,4(I1,1H-,F5.2,1X)))
  300 CONTINUE
  310 WRITE(LUNO,312)
  312 FORMAT(/// ,26X,28HMINIMUM ALLOCATION FRACTIONS//)
      WRITE(LUNO,210)
      DO 315 K=1,2
      DO 315 J=1,4
      XF(J,K)=0.
      IF(NFRAC(J,K) .EQ. 0) GO TO 315
      XF(J,K)=1./NFRAC(J,K)
  315 CONTINUE
      WRITE(LUNO,318) ((XF(J,K),J=1,4),K=1,2)
  318 FORMAT(2X,2(3X,4(F7.2,1X)))
      WRITE(LUNO,320)
  320 FORMAT(/// ,35X,11HGRID POINTS//)
      WRITE(LUNO,210)
      DO 370 I=1,11
      DO 330 K=1,2
      DO 330 J=1,4
      IF(IGRID(I,J,K) .NE. 0 .OR. I .EQ. 1) GO TO 340
  330 CONTINUE
      GO TO 400
  340 WRITE(LUNO,360) ((IGRID(I,J,K),J=1,4),K=1,2)
  360 FORMAT(3X,2(3X,4(I6,2X)))
  370 CONTINUE
  400 WRITE(LUNO,410)
  410 FORMAT(1H1//,28X,24HCAS FIREPOWER PER SORTIE//)
      WRITE(LUNO,210)
      WRITE(LUNO,420) ((CASF(J,K),J=1,4),K=1,2)
  420 FORMAT(3X,2(3X,4(F7.4,1X)))
      WRITE(LUNO,422)
  422 FORMAT(/// ,26X,28HDIVISION FIREPOWER REDUCTION,/
     134X,14HPER CAS SORTIE//)
      WRITE(LUNO,210)
      WRITE(LUNO,420) ((DFPRED(J,K),J=1,4),K=1,2)
      WRITE(LUNO,423)
```

FIGURE B-1 (cont'd)

```
 423   FORMAT(/// ,30X,20HRESIDUAL VALUE OF AN/
      123X,33HUNDAMAGED PLANE AT END OF THE WAR//)
       WRITE(LUNO,210)
       WRITE(LUNO,420) ((VALU(J,K),J=1,4),K=1,2)
       WRITE(LUNO,424)
 424   FORMAT(/// ,27X,26HFRACTION VULNERABLE TO ABA/,35X,
      110HBY MISSION//)
       WRITE(LUNO,210)
       DO 430 I=1,8
       00 425 K=1,2
       DO 425 J=1,4
       IF(1MISS(I,J,K) .NE. 0) GO TO 426
 425   CONTINUE
       GO TO 431
 426   00 427 K=1,2
       DO 427 J=1,4
       1M(J,K)=0
       XF(J,K)=0.
       IF(1MISS(1,J,K) .NE. 0) IM(J,K)=IMISS(1,J,K)
       IF(ABAF(1,J,K) .NE. 0.) XF(J,K)=ABAF(1,J,K)
 427   CONTINUE
       WRITE(LUNO,428) ((1M(J,K),XF(J,K),J=1,4),K=1,2)
 428   FORMAT(4X,2(3X,4(11,1H-,F5.3,1X)))
 430   CONTINUE
 431   IF(FEBA(1,1) .EQ. -1.) GO TO 439
       WRITE(LUNO,433)
 433   FORMAT(1H1///,33X,13HFEBA FUNCTION//)
       DO 434 1=2,28
       IF(FEBA(1,1) .EQ. 0.) GO TO 435
 434   CONTINUE
 435   1=1-1
       DO 438 K=1,I,7
       L=K+6
       WRITE(LUNO,436) (FEBA(1,J),J=K,L)
 436   FORMAT(/1X,7HF RATIO,6X,7F9.3)
       WRITE(LUNO,437) (FEBA(2,J),J=K,L)
 437   FORMAT(1X,8HMOVEMENT,5X,7F9.3)
 438   CONTINUE
 439   00 470 1=1,NSTAGE
       IF((1-1)/50*50 .NE. 1-1) GO TO 445
       WRITE(LUNO,440)
 440   FORMAT(1H1//12X,9HOBJECTIVE,2(24X,14HREINFORCEMENTS,17X)/13X,
      17HWEIGHTS,29X,6HNUMBER,48X,8HFRACTION/18X,
      12(13X,15HBLUE PLANE TYPE,12X,14HRED PLANE TYPE,1X)/1X,5HSTAGE,
      14X,4HBLUE,5X,3HRED,2(7X,19H1      2      3      4,1X),1X,
      12(7X,19H1      2      3      4,1X)/)
 445   DO 450 K=1,2
       00 450 J=1,4
       1M(J,K)=REIN(J,K,I)
       XF(J,K)=REINF(J,K,1)-1.
 450   CONTINUE
       WRITE(LUNO,460) 1,(WGHT(1,K),K=1,2),((1M(J,K),J=1,4),K=1,2),
      1((XF(J,K),J=1,4),K=1,2)
 460   FORMAT(3X,12,1X,2F8.2,2(3X,4I6),1X,2(3X,4F6.2))
 470   CONTINUE
       WRITE(LUNO,500)
 500   FORMAT(1H1,/31X,18HKILL PROBABILITIES/)
       CALL PKILLS(IHEAD(1,1),PKBDES,0.,0.)
       CALL PKILLS(IHEAD(1,2),PKESBD,0.,0.)
```

FIGURE B-1 (cont'd)

```
      CALL PKILLS(IHEAD(1,3),PKBD,0.,0.)
      CALL PKILLS(IHEAD(1,4),PKBA,0.,0.)
      WRITE(LUNO,500)
      CALL PKILLS(IHEAD(1,5),PKADES,0.,0.)
      CALL PKILLS(IHEAD(1,6),PKESAD,0.,0.)
      CALL PKILLS(IHEAD(1,7),PKAD,0.,0.)
      CALL PKILLS(IHEAD(1,8),PKAA,0.,0.)
      WRITE(LUNO,500)
      DO 600 I=1,4
      DO 600 J=1,4
      DO 600 K=1,2
      PKS(I,J,K)=PKNS(J,K)
  600 CONTINUE
      CALL PKILLS(IHEAD(1,9),PKS,0.,0.)
      DO 700 I=1,4
      DO 700 J=1,4
      DO 700 K=1,2
      PKS(I,J,K)=PKSH(J,K)
  700 CONTINUE
      CALL PKILLS(IHEAD(1,10),PKS,0.,0.)
      WRITE(LUNO,500)
      CALL SAMS(IHEAD(1,11),1.,PKFS,-2.)
      CALL SAMS(IHEAD(1,12),1.,PKFAFS,-1.)
      CALL SAMS(IHEAD(1,13),1.,-2.,PKRS)
      CALL SAMS(IHEAD(1,14),1.,PKRAFS,PKRARS)
      WRITE(LUNO,500)
      CALL SAMS(IHEAD(1,15),1.,PKBAFS,-1.)
      CALL SAMS(IHEAD(1,16),1.,PKBEFS,-1.)
      CALL SAMS(IHEAD(1,17),1.,PKAAFS,PKAARS)
      CALL SAMS(IHEAD(1,18),1.,PKAEFS,PKAERS)
      END


      SUBROUTINE PKILLS (LABL,PK,PKF,PKR)
C
C         /PKILLS/ AND /SAMS/ ARE PRINT SUBROUTINES USED IN
C         CONJUNCTION WITH /PRNTIN/.
C
      COMMON /WORKN/ NTYPE(2),NMISS(4,2),NMISST(2),NGRID(4,2),
     1IBLURO(2),NSTRAT(4,2),NFULST(2),NSTAT,NINGAM,IHPNT(32,2),
     1ITPNT(32,2),IDEM(8),NSTRTC(2),IRA(100),JRA(100),LUNI,LUNO,
     1NWORK,NSTAT2,ISINT(500,2),IDINT(100,2),INPNT(32,2)
      DIMENSION LABL(1),PK(4,4,2),PKF(4,2),PKR(4,2)
      WRITE(LUNO,100) (LABL(I),I=1,4)
  100 FORMAT(3(/),20X,4A10,///,16X,14HBLUE KILLS RED,20X,
     114HRED KILLS BLUE,//,19X,8HRED TYPE,26X,8HRED TYPE/
     12(12X,22H1       2       3       4))
      WRITE(LUNO,200) (PK(1,I,2),I=1,4),(PK(I,1,1),I=1,4)
  200 FORMAT(7X,1H1,4(2X,F5.3),6X,4(2X,F5.3))
      WRITE(LUNO,300) (PK(2,I,2),I=1,4),(PK(I,2,1),I=1,4)
  300 FORMAT(7X,1H2,4(2X,F5.3),6X,4(2X,F5.3))
      WRITE(LUNO,400) (PK(3,I,2),I=1,4),(PK(I,3,1),I=1,4)
  400 FORMAT(7X,1H3,4(2X,F5.3),6X,4(2X,F5.3))
      WRITE(LUNO,500) (PK(4,I,2),I=1,4),(PK(I,4,1),I=1,4)
  500 FORMAT(7X,1H4,4(2X,F5.3),6X,4(2X,F5.3))
      RETURN
```

FIGURE B-1 (cont'd)

```
      ENTRY SAMS
      WRITE(LUNO,600) (LABL(I),I=1,4)
  600 FORMAT(3(/),20X,4A10,////,16X,14HBLUE KILLS RED,20X,
     114HRED KILLS BLUE/)
      IF(PKF(1,1) .EQ. -2. .OR. PKR(1,1) .EQ. -2.) GO TO 650
      WRITE(LUNO,610)
  610 FORMAT(19X,8HRED TYPE,25X,9HBLUE TYPE)
      GO TO 670
  650 WRITE(LUNO,660)
  660 FORMAT(18X,9HBLUE TYPE,26X,8HRED TYPE)
  670 WRITE(LUNO,680)
  680 FORMAT(2(12X,22H1         2         3         4))
      IF(PKF(1,1) .LT. 0.) GO TO 750
      WRITE(LUNO,700) (PKF(J,2),J=1,4),(PKF(J,1),J=1,4)
  700 FORMAT(1X,7HFORWARD,4(2X,F5.3),6X,4(2X,F5.3))
      IF(PKR(1,1) .LT. 0.) RETURN
  750 WRITE(LUNO,800) (PKR(J,2),J=1,4),(PKR(J,1),J=1,4)
  800 FORMAT(2X,4HREAR,2X,4(2X,F5.3),6X,4(2X,F5.3))
      RETURN
      END
```

```
      SUBROUTINE READIN
C
C        /READIN/ READS THE INPUT PARAMETER CARDS.
C
      COMMON /INPUT/ IMISS(8,4,2),IGRID(11,4,2),LASTP,NALOC(8,4),
     1NFRAC(4,2),NSHL(2),NSTAGE,NDAPST,CASF(4,2),IPRINT(8),
     1ITITLE(6),VALU(4,2),PKBD(4,4,2),PKBDES(4,4,2),XGRID(11,4,2),
     1PKAD(4,4,2),PKADES(4,4,2),PKBA(4,4,2),PKAA(4,4,2),
     1PKESBD(4,4,2),PKESAD(4,4,2),PKSH(4,2),PKNS(4,2),REIN(4,2,100),
     1WGHT(100,2),XSORT(8,4,2),NDIV(2),OWGHT(5),NRSAM(2),NFSAM(2),
     1PKRS(4,2),PKFS(4,2),ABAF(8,4,2),DIVFP(2),PKBAFS(4,2),
     1PKAAFS(4,2),PKAARS(4,2),PKAEFS(4,2),PKAERS(4,2),PKBEFS(4,2),
     1PKFAFS(4,2),PKRAFS(4,2),PKRARS(4,2),DFPRED(4,2),FEBA(2,28),
     1REINF(4,2,100)
      COMMON /WORKN/ NTYPE(2),NMISS(4,2),NMISST(2),NGRID(4,2),
     1IBLURD(2),NSTRAT(4,2),NFULST(2),NSTAT,NINGAM,IMPNT(32,2),
     1ITPNT(32,2),IDEM(8),NSTRTC(2),IRA(100),JRA(100),LUNI,LUNO,
     1NWORK,NSTAT2,ISINT(500,2),IDINT(100,2),INPNT(32,2)
      COMMON /WORKL/ LOCST(500,2),LOCBG,LOCEG,KOCBG,KOCEG,LOCBVR,
     1LOCBVB,LOCEVR,LOCEVB,LOCBPR,LOCBPB,LOCEPR,LOCEPB,
     1KOCBVB,KOCEVR,KOCBVR,KOCEVR,JOCBVB,JOCEVB,JOCBVR,JOCEVR,
     1IOCBVB,IOCEVB,IOCBVR,IOCEVR
      COMMON /WORK/ IARRAY(25000)
      DIMENSION IDUMMY(8),KEY(35),NALOCS(33,100,2),MALOC(33)
      EQUIVALENCE (IARRAY,NALOCS),(MALOC,LASTP)
      DATA(NKEYS=30)
      DATA(IBLURD=4HBLUE,3HRED)
      DATA(LUNI=5),(LUNO=6)
      DATA(KEY=3HRUN,4HMISS,4HGRID,4HPKBD,4HPKAD,4HPKBA,
     14HPKAA,4HCASF,4HVALU,4HSTAG,4HPKBE,4HPKAE,4HNSHL,4HPKSH,
     14HPKNS,4HSTRT,4HWGHT,4HREIN,4HCWGH,4HNDIV,4HDIVF,4HDFRC,
     14HNSAM,4HPKFS,4HPKRS,4HFEBA,4HABAF,4HPKFA,4HPKRA,3HEND,5(1H ))
      DATA(REIN=800(0.)),(REINF=800(1.)),(WGHT=200(1.))
      DATA(PKBD=32(0.)),(PKBDES=32(0.)),(PKAD=32(0.)),(PKADES=32(0.))
```

FIGURE B-1 (cont'd)

```
      DATA(PKBA=32(0.)),(PKAA=32(0.)),(PKESBD=32(0.)),(PKESAD=32(0.))
      DATA(PKSH=8(0.)),(PKNS=8(0.))
      DATA(CASF=8(0.)),(VALU=8(0.)),(NSHL=2(0)),(NFRAC=8(0))
      DATA(NTYPE=2(0)),(NSTRTC=2(0)),(IGRID=88(0)),(IMISS=64(0))
      DATA(XSORT=64(0.)),(OWGHT=1.,4(0.)),(NDIV=2(0)),(NRSAM=2(0))
      DATA(NFSAM=2(0)),(PKRS=8(0.)),(PKFS=8(0.)),(ABAF=64(1.))
      DATA(DIVFP=2(0.)),(PKBAFS=8(0.)),(PKAAFS=8(0.)),(PKAARS=8(0.))
      DATA(PKAEFS=8(0.)),(PKAERS=8(0.)),(PKBEFS=8(0.)),(DFPRED=8(0.))
      DATA(FEBA=-1.,55(0.)),(PKFAFS=8(0.)),(PKRAFS=8(0.)),(PKRARS=8(0.))
      DATA(IPRINT(1)=0)
      WRITE(LUNO,70)
   70 FORMAT(1H1)
C
C        READ DATA CARD
C
  100 READ(LUN1,110) IKEY,JBR,JTP,(IDUMMY(I),I=1,8)
  110 FORMAT(A4,2A1,7A10,A4)
      IF(IPRINT(1) .EQ. 0) WRITE(LUNO,120) IKEY,JBR,JTP,
     1(IDUMMY(I),I=1,8)
  120 FORMAT(1X,A4,2A1,7A10,A4)
      DO 150 I=1,NKEYS
      IF(IKEY .EQ. KEY(I)) GO TO 200
  150 CONTINUE
      CALL ERR(2)
      GO TO 100
  200 IBR=1
      IF(JBR .EQ. 1HR) IBR=2
      DECODE(1,210,JTP) ITP
  210 FORMAT(I1)
      GO TO (300,320,340,360,380,400,420,440,460,480,500,510,
     1520,540,560,580,600,620,630,640,650,660,670,680,690,700,
     1710,720,730,800) I
C
C        RUN CARD
C
  300 DECODE(74,301,IDUMMY) (IPRINT(I),I=1,8),(ITITLE(I),I=1,6)
  301 FORMAT(4X,8I1,2X,6A10)
      GO TO 100
C
C        MISS CARD
C
  320 DECODE(74,321,IDUMMY) NFRAC(ITP,IBR),(IMISS(I,ITP,IBR),
     1I=1,8),(XSORT(I,ITP,IBR),I=1,8)
  321 FORMAT(4X,I2,1X,8(1X,I2),3X,8F5.2)
      GO TO 100
C
C        GRID CARD
C
  340 DECODE(59,341,IDUMMY) (IGRID(I,ITP,IBR),I=1,11)
  341 FORMAT(4X,11I5)
      IF(IGRID(1,ITP,IBR) .NE. 0) CALL ERR(9)
      IF(NTYPE(IBR) .LT. ITP) NTYPE(IBR)=ITP
      GO TO 100
C
C        PKBD CARD
C
  360 DECODE(49,361,IDUMMY) (PKBD(I,ITP,IBR),I=1,4),
     1(PKBDES(I,ITP,IBR),I=1,4)
  361 FORMAT(4X,4F5.3,5X,4F5.3)
```

FIGURE B-1 (cont'd)

```
      GO TO 100
C
C          PKAD CARD
C
  380 DECODE(49,361,IDUMMY) (PKAD(I,ITP,IBR),I=1,4),
     1(PKADES(I,ITP,IBR),I=1,4)
      GO TO 100
C
C          PKBA CARD
C
  400 DECODE(34,401,IDUMMY) (PKBA(I,ITP,IBR),I=1,4),PKBAFS(ITP,IBR)
  401 FORMAT(4X,4F5.3,5X,F5.3)
      GO TO 100
C
C          PKAA CARD
C
  420 DECODE(39,421,IDUMMY) (PKAA(I,ITP,IBR),I=1,4),PKAAFS(ITP,IBR),
     1PKAARS(ITP,IBR)
  421 FORMAT(4X,4F5.3,5X,4F5.3)
      GO TO 100
C
C          CASF CARD
C
  440 DECODE(24,441,IDUMMY) (CASF(I,IBR),I=1,4)
  441 FORMAT(4X,4F5.4)
      GO TO 100
C
C          VALU CARD
C
  460 DECODE(24,461,IDUMMY) (VALU(I,IBR),I=1,4)
  461 FORMAT(4X,4F5.0)
      GO TO 100
C
C          STAG CARD
C
  480 DECODE(10,481,IDUMMY) NSTAGE,NDAPST
  481 FORMAT(2X,2(2X,I2))
      GO TO 100
C
C          PKBE CARD
C
  500 DECODE(34,401,IDUMMY) (PKESBD(I,ITP,IBR),I=1,4),PKBEFS(ITP,IBR)
      GO TO 100
C
C          PKAE CARD
C
  510 DECODE(39,421,IDUMMY) (PKESAD(I,ITP,IBR),I=1,4),PKAEFS(ITP,IBR),
     1PKAERS(ITP,IBR)
      GO TO 100
C
C          NSHL CARD
C
  520 DECODE( 9,521,IDUMMY) NSHL(IBR)
  521 FORMAT(4X,I5)
      GO TO 100
C
C          PKSH CARD
C
  540 DECODE(24,541,IDUMMY) (PKSH(I,IBR),I=1,4)
  541 FORMAT(4X,4F5.3)
```

FIGURE B-1 (cont'd)

```
      GO TO 100
C
C          PKNS CARD
C
  560 DECODE(24,541,IDUMMY) (PKNS(I,IBR),I=1,4)
      GO TO 100
C
C          STRT CARD
C
  580 DECODE(74,581,IDUMMY) (MALOC(I),I=1,33)
  581 FORMAT(I2,4(2X,8A2))
      NSTRTC(IBR)=NSTRTC(IBR)+1
      DO 583 I=2,33
      IF(MALOC(I) .EQ. 2H * .OR. MALOC(I) .EQ. 2H**) MALOC(I)=2H-1
      DECODE(2,582,MALOC(I)) IHOLD
  582 FORMAT(I2)
      MALOC(I)=IHOLD
  583 CONTINUE
      J=NSTRTC(IBR)
      DO 584 I=1,33
      NALOCS(I,J,IBR)=MALOC(I)
  584 CONTINUE
      GO TO 100
C
C          WGHT CARD
C
  600 DECODE(9,601,IDUMMY) J,WGHT(J,IBR)
  601 FORMAT(I2,2X,F5.0)
      GO TO 100
C
C          REIN CARD
C
  620 DECODE(49,621,IDUMMY) J,(REIN(I,IBR,J),I=1,4),
     1(REINF(I,IBR,J),I=1,4)
  621 FORMAT(I2,2X,4F5.0,5X,4F5.0)
      DO 625 I=1,4
      REINF(I,IBR,J)=REINF(I,IBR,J)+1.
  625 CONTINUE
      GO TO 100
C
C          OWGHT CARD
C
  630 DECODE(29,631,IDUMMY) (OWGHT(I),I=1,5)
  631 FORMAT(4X,5F5.0)
      GO TO 100
C
C          NDIV CARD
C
  640 DECODE(9,521,IDUMMY) NDIV(IBR)
      GO TO 100
C
C          DIVF CARD
C
  650 DECODE(9,651,IDUMMY) DIVFP(IBR)
  651 FORMAT(4X,F5.0)
      GO TO 100
C
C          DFRC CARD
C
```

FIGURE B-1 (cont'd)

```
  660   DECODE(24,441,IDUMMY) (DFPRED(I,IBR),I=1,4)
        GO TO 100
C
C           NSAM CARD
C
  670   DECODE(14,671,IDUMMY) NFSAM(IBR),NRSAM(IBR)
  671   FORMAT(4X,2I5)
        GO TO 100
C
C           PKFS CARD
C
  680   DECODE(24,541,IDUMMY) (PKFS(I,IBR),I=1,4)
        GO TO 100
C
C           PKRS CARD
C
  690   DECODE(24,541,IDUMMY) (PKRS(I,IBR),I=1,4)
        GO TO 100
C
C           FEBAM CARD
C
  700   ILO=(ITP-1)*7+1
        IHI=ILO+6
        DECODE(74,701,IDUMMY) ((FEBA(I,J),I=1,2),J=ILO,IHI)
  701   FORMAT(4X,14F5.0)
        GO TO 100
C
C           ABAF CARD
C
  710   DECODE(44,711,IDUMMY) (ABAF(I,ITP,IBR),I=1,8)
  711   FORMAT(4X,8F5.0)
        GO TO 100
C
C           PKFA CARD
C
  720   DECODE(9,721,IDUMMY) PKFAFS(ITP,IBR)
  721   FORMAT(4X,F5.3)
        GO TO 100
C
C           PKRA CARD
C
  730   DECODE(14,731,IDUMMY) PKRAFS(ITP,IBR),PKRARS(ITP,IBR)
  731   FORMAT(4X,2F5.3)
        GO TO 100
C
C           END CARD
C
  800   DO 810 K=1,2
        J=NSTRTC(K)
        DO 810 I=1,J
        BUFFER OUT (9,1) (NALOCS(1,I,K),NALOCS(33,I,K))
        IF(UNIT(9)) 810,807,807
  807   CALL ERR(21)
  810   CONTINUE
        RETURN
        END
```

FIGURE B-1 (cont'd)

```
      SUBROUTINE STRAT
C
C          /STRAT/ GENERATES THE PURE STRATEGIES FOR A PARTICULAR PLANE
C          TYPE RESULTING FROM A SPECIFIED ALLOCATION OF A MINIMUM
C          ALLOCATION FRACTION TO MISSIONS.
C
      COMMON /INPUT/ IMISS(8,4,2),IGRID(11,4,2),LASTP,NALOC(8,4),
     1NFRAC(4,2),NSHL(2),NSTAGE,NDAPST,CASF(4,2),IPRINT(8),
     1ITITLE(6),VALU(4,2),PKBD(4,4,2),PKBDES(4,4,2),XGRID(11,4,2),
     1PKAD(4,4,2),PKADES(4,4,2),PKBA(4,4,2),PKAA(4,4,2),
     1PKESBD(4,4,2),PKESAD(4,4,2),PKSH(4,2),PKNS(4,2),REIN(4,2,100),
     1WGHT(100,2),XSORT(8,4,2),NDIV(2),OWGHT(5),NRSAM(2),NFSAM(2),
     1PKRS(4,2),PKFS(4,2),ABAF(8,4,2),DIVFP(2),PKBAFS(4,2),
     1PKAAFS(4,2),PKAARS(4,2),PKAEFS(4,2),PKAERS(4,2),PKBEFS(4,2),
     1PKFAFS(4,2),PKRAFS(4,2),PKRARS(4,2),DFPRED(4,2),FEBA(2,28),
     1REINF(4,2,100)
      COMMON /WORKN/ NTYPE(2),NMISS(4,2),NMISST(2),NGRID(4,2),
     1IBLURD(2),NSTRAT(4,2),NFULST(2),NSTAT,NINGAM,IMPNT(32,2),
     1ITPNT(32,2),IDEM(8),NSTRTC(2),IRA(100),JRA(100),LUNI,LUNO,
     1NWORK,NSTAT2,ISINT(500,2),IDINT(100,2),INPNT(32,2)
      COMMON /WORKL/ LOCST(500,2),LOCBG,LOCEG,KOCBG,KOCEG,LOCBVR,
     1LOCBVB,LOCEVR,LOCEVB,LOCBPR,LOCBPB,LOCEPR,LOCEPB,
     1KOCBVB,KOCEVB,KOCBVR,KOCEVR,JOCBVB,JOCEVB,JOCBVR,JOCEVR,
     1IOCBVB,IOCEVB,IOCBVR,IOCEVR
      COMMON /WORK/ SPACER(18600),STRATS(8,200,4)
C
C          IF NWORK IS CHANGED IN INIT, THE LENGTH OF SPACER MUST
C          BE SET EQUAL TO NWORK-6400.
C
      COMMON /SPARM/ MFRAC,MMISS,IBR,ITYPE,NSTOR(8),IPNT(8)
      DIMENSION MSTOR(8)
      NSTRAT(ITYPE,IBR)=0
      MISSN=NMISS(ITYPE,IBR)
      FRACN=NFRAC(ITYPE,IBR)
      IF(MMISS .NE. 0) GO TO 200
      DO 100 I=1,MISSN
      STRATS(I,1,ITYPE)=NSTOR(I)/FRACN
  100 CONTINUE
      NSTRAT(ITYPE,IBR)=1
      RETURN
  200 IHI1=MFRAC+1
      ILO1=1
      IF(MMISS .EQ. 1) ILO1=IHI1
      DO 400 I1=ILO1,IHI1
      MSTOR(1)=I1-1
      IHI2=IHI1-MSTOR(1)
      ILO2=1
      IF(MMISS .EQ. 2) ILO2=IHI2
      DO 400 I2=ILO2,IHI2
      MSTOR(2)=I2-1
      IHI3=IHI2-MSTOR(2)
      ILO3=1
      IF(MMISS .EQ. 3) ILO3=IHI3
      DO 400 I3=ILO3,IHI3
      MSTOR(3)=I3-1
      IHI4=IHI3-MSTOR(3)
      ILO4=1
      IF(MMISS .EQ. 4) ILO4=IHI4
      DO 400 I4=ILO4,IHI4
```

FIGURE B-1 (cont'd)

```
      MSTOR(4)=I4-1
      IHI5=IHI4-MSTOR(4)
      IL05=1
      IF(MMISS .EQ. 5) IL05=IHI5
      DO 400 I5=IL05,IHI5
      MSTOR(5)=I5-1
      IHI6=IHI5-MSTOR(5)
      IL06=1
      IF(MMISS .EQ. 6) IL06=IHI6
      DO 400 I6=IL06,IHI6
      MSTOR(6)=I6-1
      IHI7=IHI6-MSTOR(6)
      IL07=1
      IF(MMISS .EQ. 7) IL07=IHI7
      DO 400 I7=IL07,IHI7
      MSTOR(7)=I7-1
      IHI8=IHI7-MSTOR(7)
      IL08=1
      IF(MMISS .EQ. 8) IL08=IHI8
      DO 400 I8=IL08,IHI8
      MSTOR(8)=I8-1
      NSTRAT(ITYPE,IBR)=NSTRAT(ITYPE,IBR)+1
      DO 300 I=1,MMISS
      J=IPNT(I)
      NSTOR(J)=MSTOR(I)
  300 CONTINUE
      J=NSTRAT(ITYPE,IBR)
      DO 350 I=1,MISSN
      STRATS(I,J,ITYPE)=NSTOR(I)/FRACN
  350 CONTINUE
  400 CONTINUE
      IF(NSTRAT(ITYPE,IBR) .GT. 200) CALL ERR(3)
      RETURN
      END



      SUBROUTINE TIMER
C
C         /TIMER/ USES AN EMPIRICAL FORMULA TO ESTIMATE THE
C         EXECUTION TIME REQUIRED FOR THE CURRENT RUN.
C
      COMMON /INPUT/ IMISS(8,4,2),IGRID(11,4,2),LASTP,NALOC(8,4),
     1NFRAC(4,2),NSHL(2),NSTAGE,NDAPST,CASF(4,2),IPRINT(8),
     1ITITLE(6),VALU(4,2),PKBD(4,4,2),PKBDES(4,4,2),XGRID(11,4,2),
     1PKAD(4,4,2),PKADES(4,4,2),PKBA(4,4,2),PKAA(4,4,2),
     1PKESBD(4,4,2),PKESAD(4,4,2),PKSH(4,2),PKNS(4,2),REIN(4,2,100),
     1WGHT(100,2),XSORT(8,4,2),NDIV(2),OWGHT(5),NRSAM(2),NFSAM(2),
     1PKRS(4,2),PKFS(4,2),ABAF(8,4,2),DIVFP(2),PKBAFS(4,2),
     1PKAAFS(4,2),PKAARS(4,2),PKAEFS(4,2),PKAERS(4,2),PKBEFS(4,2),
     1PKFAFS(4,2),PKRAFS(4,2),PKRARS(4,2),DFPRED(4,2),FEBA(2,28),
     1REINF(4,2,100)
      COMMON /WORKN/ NTYPE(2),NMISS(4,2),NMISST(2),NGRID(4,2),
     1IBLURD(2),NSTRAT(4,2),NFULST(2),NSTAT,NINGAM,IMPNT(32,2),
     1ITPNT(32,2),IDEM(8),NSTRTC(2),IRA(100),JRA(100),LUNI,LUNO,
     1NWORK,NSTAT2,ISINT(500,2),IDINT(100,2),INPNT(32,2)
```

FIGURE B-1 (cont'd)

```
      COMMON /WORKL/ LOCST(500,2),LOCBG,LOCEG,KOCBG,KOCEG,LOCBVR,
     1LOCBVB,LOCEVR,LOCEVB,LOCBPR,LOCBPB,LOCEPR,LOCEPB,
     1KOCBVB,KOCEVB,KOCBVR,KOCEVR,JOCBVB,JOCEVB,JOCBVR,JOCEVR,
     1IOCBVB,IOCEVB,IOCBVR,IOCEVR
      COMMON /SINTVL/ IRLO(500),IRHI(500)
      DIMENSION IFUNC(4),TIMEC(4),TIMEI(4)
      DATA(V=.12),(W=2.0E-3),(X=1.2E-5),(Y=4.4E-4),(Z=2.0E-4)
      DATA(IFUNC=5HSETUP,7HBATTLES,5HGAMES,5HTOTAL)
      IHI1=NFULST(1)
      IHI2=NFULST(2)
      TIMEC(1)=2.0
      TIMEI(1)=2.0
      NTP=NTYPE(1)+NTYPE(2)
      NPNTS=1
      DO 100 I=1,NTP
      NPNTS=NPNTS*2
  100 CONTINUE
      NBATLS=0
      TIMEC(3)=0.
      DO 200 I=1,IHI1
      NBATLS=NBATLS+IRHI(I)-IRLO(I)+1
  200 CONTINUE
      TIMEC(2)=NSTAT*NDAPST*NBATLS*W
      TIMEI(2)=NSTAT*V
      DO 500 I=1,NSTAGE
      ICNT=0
      DO 300 J=1,IHI1
      IF(ISINT(J,1) .NE. IDINT(I,1)) GO TO 300
      ICNT=ICNT+1
  300 CONTINUE
      JCNT=0
      DO 400 J=1,IHI2
      IF(ISINT(J,2) .NE. IDINT(I,2)) GO TO 400
      JCNT=JCNT+1
  400 CONTINUE
      TIMEC(3)=TIMEC(3)+ICNT*JCNT*(Y+NTP*X*NPNTS)+Z*(ICNT+JCNT)
  500 CONTINUE
      TIMEC(3)=NSTAT*TIMEC(3)
      TIMEI(3)=NSTAGE*NSTAT*V
      IF(IPRINT(4) .EQ. 1) TIMEI(2)=2.*TIMEI(2)
      IF(IPRINT(4) .EQ. 2) TIMEC(2)=0.
      TIMEC(4)=TIMEC(1)+TIMEC(2)+TIMEC(3)
      TIMEI(4)=TIMEI(1)+TIMEI(2)+TIMEI(3)
      WRITE(LUNO,600)
  600 FORMAT(1H1///10X,40HCDC 6600 TIME ESTIMATES FOR CURRENT RUN ,
     19H(SECONDS)//16X,8HFUNCTION,8X,8HCPU TIME,3X,8HI/O TIME/)
      DO 700 I=1,3
      WRITE(LUNO,650) IFUNC(I),TIMEC(I),TIMEI(I)
  650 FORMAT(16X,A7,9X,F8.1,3X,F8.1)
  700 CONTINUE
      WRITE(LUNO,750) IFUNC(4),TIMEC(4),TIMEI(4)
  750 FORMAT(/16X,A7,9X,F8.1,3X,F8.1///)
      RETURN
      END
```

FIGURE B-1 (cont'd)

```
      SUBROUTINE TRIALS
C
C        /TRIALS/ COMPUTES THE MAXMIN AND MINMAX BOUNDS AND PERFORMS
C        THE FORWARD EVALUATION FOR A TRIAL AIRWAR OF SPECIFIED
C        LENGTH BEGINNING WITH A SPECIFIED NUMBER OF BLUE AND RED
C        PLANES.  /TRIALS/ COMPUTES THESE VALUES USING THE OPTIMAL
C        STRATEGIES AND VALUES PREVIOUSLY DETERMINED BY /GAMES/.
C
      COMMON /INPUT/ IMISS(8,4,2),IGRID(11,4,2),LASTP,NALOC(8,4),
     1NFRAC(4,2),NSHL(2),NSTAGE,NDAPST,CASF(4,2),IPRINT(8),
     1ITITLE(6),VALU(4,2),PKBD(4,4,2),PKBDES(4,4,2),XGRID(11,4,2),
     1PKAD(4,4,2),PKADES(4,4,2),PKBA(4,4,2),PKAA(4,4,2),
     1PKESBD(4,4,2),PKESAD(4,4,2),PKSH(4,2),PKNS(4,2),REIN(4,2,100),
     1WGHT(100,2),XSORT(8,4,2),NDIV(2),OWGHT(5),NRSAM(2),NFSAM(2),
     1PKRS(4,2),PKFS(4,2),ABAF(8,4,2),DIVFP(2),PKBAFS(4,2),
     1PKAAFS(4,2),PKAARS(4,2),PKAEFS(4,2),PKAERS(4,2),PKBEFS(4,2),
     1PKFAFS(4,2),PKRAFS(4,2),PKRARS(4,2),DFPRED(4,2),FEBA(2,28),
     1REINF(4,2,100)
      COMMON /WORKN/ NTYPE(2),NMISS(4,2),NMISST(2),NGRID(4,2),
     1IBLURD(2),NSTRAT(4,2),NFULST(2),NSTAT,NINGAM,IMPNT(32,2),
     1ITPNT(32,2),IDEM(8),NSTRTC(2),IRA(100),JRA(100),LUNI,LUNO,
     1NWORK,NSTAT2,ISINT(500,2),IDINT(100,2),INPNT(32,2)
      COMMON /WORKL/ LOCST(500,2),LOCBG,LOCEG,KOCBG,KOCEG,LOCBVR,
     1LOCBVB,LOCEVR,LOCEVB,LOCBPR,LOCBPB,LOCEPR,LOCEPB,
     1KOCBVB,KOCEVB,KOCBVR,KOCEVR,JOCBVB,JOCEVB,JOCBVR,JOCEVR,
     1IOCBVB,IOCEVB,IOCBVR,IOCEVR
      COMMON /BPARM/ CNP(4,2),IB,IR,XNP(9,4,2),OBJEC(2,5)
      COMMON /INTERP/ BETA(2,8),IBETA(2,8),I1,I2,I3,I4,I5,I6,I7,I8,
     1IHI1,IHI2,IHI3,IHI4,IHI5,IHI6,IHI7,IHI8
      COMMON /WORK/ XARRAY(25000)
      DIMENSION IOPTN(5),IOUT(10),JOUT(4,2),JOPTN(5),IARRAY(1)
      DIMENSION IPLAL(100,2),XOBJF(3,100),TXOBJF(3)
      EQUIVALENCE (XARRAY,IARRAY)
      COMMON /ROUND/ JDEX(4,2,2)
      DIMENSION MVEC(1),NVEC(1)
      EQUIVALENCE (MVEC,JDEX(1,1,1)),(NVEC,JDEX(1,1,2))
      DATA(JOPTN=5(0))
      NTRIAL=0
  100 READ(LUNI,110) KEY,MSTAGE,(IOPTN(I),I=1,5),((CNP(I,K),I=1,4),
     1K=1,2)
  110 FORMAT(A5,5X,I2,2X,5I1,1X,2(4F5.0,5X))
      IF(EOF(LUNI)) 990,120
  120 IF(KEY .EQ. 5HTRIAL) GO TO 200
      IF(KEY .EQ. 5HFINIS) GO TO 990
      CALL ERR(17)
      GO TO 100
  200 IF(MSTAGE .LE. NSTAGE) GO TO 240
      CALL ERR(18)
      GO TO 100
  240 DO 250 K=1,2
      DO 250 I=1,4
      IUP=NGRID(I,K)
      IF(CNP(I,K) .GT. XGRID(IUP,I,K)) GO TO 270
  250 CONTINUE
      GO TO 300
  270 CALL ERR(19)
      GO TO 100
  300 REWIND 7
```

FIGURE B-1 (cont'd)

```
      REWIND 8
      REWIND 9
      TOBJF=0.
      DO 305 K=1,2
      DO 305 I=1,4
      JOUT(I,K)=CNP(I,K)+.5
  305 CONTINUE
      NTRIAL=NTRIAL+1
      NSKIP=NSTAGE-MSTAGE
      IF(NSKIP .EQ. 0) GO TO 400
      DO 350 I=1,NSKIP
      BUFFER IN (7,1) (IARRAY(LOCBVB),IARRAY(LOCBVB))
      IF(UNIT(7)) 320,310,310
  310 CALL ERR(14)
  320 BUFFER IN (8,1) (IARRAY(LOCBPB),IARRAY(LOCBPB))
      IF(UNIT(8)) 350,340,340
  340 CALL ERR(15)
  350 CONTINUE
  400 BUFFER IN (7,1) (IARRAY(LOCBVB),IARRAY(LOCEVR))
      IF(UNIT(7)) 410,405,405
  405 CALL ERR(14)
C
C         COMPUTE MAXMIN AND MINMAX
C
  410 DO 420 K=1,16
      MVEC(K)=0
  420 CONTINUE
      DO 460 K=1,2
      L=3-K
      IUP=NTYPE(L)
      DO 460 J=1,IUP
      N=1
      DO 440 M=1,11
      IF(CNP(J,L)-XGRID(M,J,L)) 450,445,440
  440 CONTINUE
      M=11
  445 N=0
  450 M=M-1
      JDEX(J,L,K)=M
      JDEX(J,L,L)=M-N
  460 CONTINUE
      IBS=ISTATE(MVEC)-1
      IRS=ISTATE(NVEC)-1
      XMIN=XARRAY(LOCBVB+IBS)
      XMAX=XARRAY(LOCBVR+IRS)
C
C         PERFORM FORWARD EVALUATION FOR SPECIFIED TRIAL
C
      DO 700 M=1,MSTAGE
      MSLOC=M+NSKIP
      NSLOC=MSLOC+1
      BUFFER IN (8,1) (IARRAY(LOCBPB),IARRAY(LOCEPR))
      IF(UNIT(8)) 510,505,505
  505 CALL ERR(15)
  510 CALL BETAS
      DO 600 N=1,8
      MVEC(N)=IBETA(1,N)
      IF(BETA(1,N) .LT. BETA(2,N)) MVEC(N)=IBETA(2,N)
  600 CONTINUE
```

FIGURE B-1 (cont'd)

```
      IS=ISTATE(MVLC)-1
      IB=IARRAY(LOCBPB+IS)
      IR=IARRAY(LOCBPR+IS)
      IPLAL(M,1)=LOCST(IB,1)
      IPLAL(M,2)=LOCST(IR,2)
      CALL BATTLE
      BOBJ=0.
      ROBJ=0.
      DO 605 I=1,3
      BOBJ=BOBJ+OBJEC(1,I)*OWGHT(I)
      ROBJ=ROBJ+OBJEC(2,I)*OWGHT(I)
  605 CONTINUE
      OBJF=BOBJ*WGHT(MSLOC,1)-ROBJ*WGHT(MSLOC,2)
      TOBJF=TOBJF+OBJF
      IF(IOPTN(I) .NE. 0) GO TO 630
      ICNT=0
      DO 610 K=1,2
      DO 610 I=1,4
      ICNT=ICNT+1
      IOUT(ICNT)=CNP(I,K)+.5
      CNP(I,K)=REINF(I,K,NSLOC)*CNP(I,K)+REIN(I,K,NSLOC)
  610 CONTINUE
      IOUT(9)=OBJF+SIGN(.5,OBJF)
      IOUT(10)=TOBJF+SIGN(.5,TOBJF)
      BUFFER OUT (9,1) (IOUT,IOUT(10))
      IF(UNIT(9)) 630,620,620
  620 CALL ERR(20)
  630 IF(IOPTN(3) .NE. 0) GO TO 700
      DO 640 I=1,3
      XOBJF(I,M)=OBJEC(1,I)*WGHT(MSLOC,1)-OBJEC(2,I)*WGHT(MSLOC,2)
  640 CONTINUE
  700 CONTINUE
C
C         WRITE DESIGNATED OUTPUT FOR CURRENT TRIAL
C
      IF((NTRIAL-1)/25*25 .EQ. NTRIAL-1) GO TO 720
      DO 710 I=1,3
      IF(JOPTN(I) .EQ. 0) GO TO 720
  710 CONTINUE
      GO TO 730
  720 WRITE(LUNO,725)
  725 FORMAT(1H1,//8X,6HNUMBER,7X,26HNUMBER OF PLANES AVAILABLE,27X
     1,6HMAXMIN/,1X,5HTRIAL,4X,2HOF,5X,16H----- BLUE -----,4X,
     116H----- RED ------,4X,4HBLUE,5X,3HRED,7X,2HVS,/,IX,
     113HNUMBER STAGES,3X,I6HI     2    3    4,4X,
     116H1     2    3    4,3X,6HMAXMIN,3X,6HMINMAX,3X,6HMINMAX)
  730 IMIN=XMIN+SIGN(.5,XMIN)
      IMAX=XMAX+SIGN(.5,XMAX)
      IOBJF=TOBJF+SIGN(.5,TOBJF)
      DO 735 I=1,5
      JOPTN(I)=IOPTN(I)
  735 CONTINUE
      WRITE(LUNO,740) NTRIAL,MSTAGE,((JOUT(I,K),I=1,4),K=1,2),IMIN,
     1IMAX,IOBJF
  740 FORMAT(/2X,I3,5X,I2,2X,8I5,2(I8,1X),I8)
      IF(IOPTN(1) .NE. 0) GO TO 850
      REWIND 9
      DO 840 I=I,MSTAGE
      IF((I-I)/50*50 .EQ. I-I) WRITE(LUNO,800) NTRIAL
```

FIGURE B-1 (cont'd)

```
 800   FORMAT(1H1,//21X,12HTRIAL NUMBER,I5//12X,14HNUMBER OF BLUE,7X,
      113HNUMBER OF RED,8X,6HMAXMIN,/,1X,5HSTAGE,5X,
      12(16HPLANES AVAILABLE,5X),3X,2HVS,/,1X,6HNUMBER,4X,
      12(16H1     2     3     4,5X),1X,6HMINMAX,4X,5HTOTAL/)
       BUFFER IN (9,1) (IOUT,IOUT(10))
       IF(UNIT(9)) 810,805,805
 805   CALL ERR(20)
 810   WRITE(LUNO,820) I,(IOUT(J),J=1,10)
 820   FORMAT(1X,I4,2X,2(1X,4I5),2X,2I9)
 840   CONTINUE
 850   IF(IOPTN(2) .NE. 0) GO TO 900
       DO 890 K=1,2
       IUP=NMISST(K)
       DO 890 I=1,MSTAGE
       IF((I-1)/50*50 .NE. I-1) GO TO 865
       WRITE(LUNO,860) NTRIAL,IBLURD(K),
      1(ITPNT(J,K),IMPNT(J,K),J=1,IUP)
 860   FORMAT(1H1,//21X,12HTRIAL NUMBER,I5,//13X,
      131HPLANE ALLOCATION FRACTIONS FOR ,A4,//,1X,5HSTAGE,16X,
      118HPLANE TYPE/MISSION,/,1X,6HNUMBER,5X,
      13(12(I1,1H/,I1,2X)/12X))
       WRITE(LUNO,862)
 862   FORMAT(1H )
 865   ILO=IPLAL(I,K)
       ITP=ILO+IUP-1
       WRITE(LUNO,880) I,(XARRAY(J),J=ILO,ITP)
 880   FORMAT(1X,I4,5X,3(12F5.2,/,10X))
 890   CONTINUE
 900   IF(IOPTN(3) .NE. 0) GO TO 100
       DO 905 I=1,3
       TXOBJF(I)=0.
 905   CONTINUE
       DO 930 I=1,MSTAGE
       IF((I-1)/50*50 .EQ. I-1) WRITE(LUNO,910) NTRIAL
 910   FORMAT(1H1//21X,12HTRIAL NUMBER,I5,//12X,8HBLUE-RED,15X,
      18HBLUE-RED/1X,5HSTAGE,8X,3HCAS,18X,8HGRND+AIR,17X,4HFEBA/
      11X,6HNUMBER,5X,9HFIREPOWER,4X,5HTOTAL,5X,9HFIREPOWER,4X,
      15HTOTAL,5X,8HMOVEMENT,5X,5HTOTAL/)
       DO 915 J=1,3
       TXOBJF(J)=TXOBJF(J)+XOBJF(J,I)
       K=2*J
       IOUT(K-1)=XOBJF(J,I)+SIGN(.5,XOBJF(J,I))
       IOUT(K)=TXOBJF(J)+SIGN(.5,TXOBJF(J))
 915   CONTINUE
       WRITE(LUNO,920) I,(IOUT(J),J=1,6)
 920   FORMAT(1X,I4,2X,3(3X,2I10))
 930   CONTINUE
       GO TO 100
 990   RETURN
       END
```

FIGURE B-2

ATACM2 LISTING

```
      PROGRAM ATACM2 (OUTPUT,TAPE4=65,TAPE5,TAPE6=OUTPUT,TAPE7=65,
     1TAPE8=65,TAPE9=65)
C
C         /ATACM2/ AND ASSOCIATED SUBROUTINES ARE USED TO EVALUATE
C         TRIAL AIRWARS USING THE OPTIMAL STRATEGIES AND GAME VALUES
C         INPUT FROM A TRIAL-TAPE WRITTEN DURING A PREVIOUS RUN
C         OF /ATACM1/.
C
      COMMON /INPUT/ IMISS(8,4,2),IGRID(11,4,2),LASTP,NALOC(8,4),
     1NFRAC(4,2),NSHL(2),NSTAGE,NDAPST,CASF(4,2),IPRINT(8),
     1ITITLE(6),VALU(4,2),PKBD(4,4,2),PKBDES(4,4,2),XGRID(11,4,2),
     1PKAD(4,4,2),PKADES(4,4,2),PKBA(4,4,2),PKAA(4,4,2),
     1PKESBD(4,4,2),PKESAD(4,4,2),PKSH(4,2),PKNS(4,2),REIN(4,2,100),
     1WGHT(100,2),XSORT(8,4,2),NDIV(2),OWGHT(5),NRSAM(2),NFSAM(2),
     1PKRS(4,2),PKFS(4,2),ABAF(8,4,2),DIVFP(2),PKBAFS(4,2),
     1PKAAFS(4,2),PKAARS(4,2),PKAEFS(4,2),PKAERS(4,2),PKBEFS(4,2),
     1PKFAFS(4,2),PKRAFS(4,2),PKRARS(4,2),DFPRED(4,2),FEBA(2,28),
     1REINF(4,2,100)
      COMMON /WORKN/ NTYPE(2),NMISS(4,2),NMISST(2),NGRID(4,2),
     1IBLURD(2),NSTRAT(4,2),NFULST(2),NSTAT,NINGAM,IMPNT(32,2),
     1ITPNT(32,2),IOEM(8),NSTRTC(2),IRA(100),JRA(100),LUNI,LUNO,
     1NWORK,NSTAT2,ISINT(500,2),IDINT(100,2),INPNT(32,2)
      COMMON /WORKL/ LOCST(500,2),LOCBG,LOCEG,KOCBG,KOCEG,LOCBVR,
     1LOCBVB,LOCEVR,LOCEVB,LOCBPR,LOCBPB,LOCEPR,LOCEPB,
     1KOCBVB,KOCEVB,KOCBVR,KOCEVR,JOCBVB,JOCEVB,JOCBVR,JOCEVR,
     1IOCBVB,IOCEVB,IOCBVR,IOCEVR
      COMMON /WORK/ XARRAY(25000)
      COMMON /ERROR/ IERR
      DIMENSION INPUTZ(2680),WORKNZ(1640),WORKLZ(1024),WORKZ(25000)
      DIMENSION IARRAY(10000)
      DIMENSION JGRID(11,8),JPRINT(8)
      EQUIVALENCE (IMISS,INPUTZ),(NTYPE,WORKNZ),(LOCST,WORKLZ)
      EQUIVALENCE (IGRID,JGRID)
      EQUIVALENCE (XARRAY,IARRAY,WORKZ)
      BUFFER IN (4,1) (INPUTZ,INPUTZ(2680))
      IF(UNIT(4)) 200,150,150
  150 CALL ERR(22)
  200 BUFFER IN (4,1) (WORKNZ,WORKNZ(1640))
      IF(UNIT(4)) 300,150,150
  300 BUFFER IN (4,1) (WORKLZ,WORKLZ(1024))
      IF(UNIT(4)) 400,150,150
  400 BUFFER IN (4,1) (WORKZ,WORKZ(NWORK))
      IF(UNIT(4)) 410,150,150
  410 DO 480 I=1,NSTAGE
      BUFFER IN (4,1) (IARRAY(LOCBVB),IARRAY(LOCEVR))
      IF(UNIT(4)) 415,450,450
  415 BUFFER OUT (7,1) (IARRAY(LOCBVB),IARRAY(LOCEVR))
      IF(UNIT(7)) 420,460,460
  420 BUFFER IN (4,1) (IARRAY(LOCBPB),IARRAY(LOCEPR))
      IF(UNIT(4)) 425,450,450
  425 BUFFER OUT (8,1) (IARRAY(LOCBPB),IARRAY(LOCEPR))
      IF(UNIT(8)) 480,470,470
  450 CALL ERR(22)
  460 CALL ERR(12)
  470 CALL ERR(13)
  480 CONTINUE
      READ(LUNI,490) IKEY,(JPRINT(I),I=1,8)
  490 FORMAT(A3,7X,8I1)
      IF(IKEY .NE. 3HRUN) CALL ERR(16)
```

FIGURE B-2 (cont'd)

```
  500   IF(JPRINT(1) .EQ. 0) CALL PRNTIN
        IF(JPRINT(2) .NE. 0) GO TO 700
        DO 560 K=1,2
        IHI3=NMISST(K)
        IHI1=NFULST(K)
        DO 560 I=1,IHI1
        ILO2=LOCST(I,K)
        IHI2=ILO2+NMISST(K)-1
        IF((I-1)/50*50 .NE. I-1) GO TO 538
        WRITE(LUNO,530) IBLURD(K),(ITFNT(J,K),IMPNT(J,K),J=1,IHI3)
  530   FORMAT(1H1///,17X,A5,15HPURE STRATEGIES//,6H STRAT,4X,4HLAST,
       112X,18HPLANE TYPE/MISSION/,7H NUMBER,3X,5HSTAGE,5X,
       14(10(I1,1H/,I1,2X)/20X))
        WRITE(LUNO,535)
  535   FORMAT(1H )
  538   WRITE(LUNO,540) I,ISINT(I,K),(XARRAY(J),J=ILO2,IHI2)
  540   FORMAT(1X,I5,5X,I3,4X,4(10F5.2,/,18X))
  560   CONTINUE
  700   IF(JPRINT(3) .NE. 0) GO TO 800
        ICNT=0
        NG1=NGRID(1,1)
        NG2=NGRID(2,1)
        NG3=NGRID(3,1)
        NG4=NGRID(4,1)
        NG5=NGRID(1,2)
        NG6=NGRID(2,2)
        NG7=NGRID(3,2)
        NG8=NGRID(4,2)
        DO 730 I1=1,NG1
        DO 730 I2=1,NG2
        DO 730 I3=1,NG3
        DO 730 I4=1,NG4
        DO 730 I5=1,NG5
        DO 730 I6=1,NG6
        DO 730 I7=1,NG7
        DO 730 I8=1,NG8
        ICNT=ICNT+1
        IF((ICNT-1)/50*50 .EQ. ICNT-1) WRITE(LUNO,710)
  710   FORMAT(1H1,//18X,27HLIST OF ALL POSSIBLE STATES,//6H STATE,
       115X,4HBLUE,27X,3HRED,/7H NUMBER,7X,2(19H1      2     3     4,
       111X)/)
        WRITE(LUNO,720) ICNT,JGRID(I1,1),JGRID(I2,2),JGRID(I3,3),
       1JGRID(I4,4),JGRID(I5,5),JGRID(I6,6),JGRID(I7,7),JGRID(I8,8)
  720   FORMAT(1X,I5,4X,2(4I6,6X))
  730   CONTINUE
  800   CALL TRIALS
        END
```

In addition, ATACM2 uses subroutines

BATTLE
BETAS
ERR
ISTATE
PRNTIN
TRIALS
PKILLS

all of which are listed under ATACM1.

TABLE B-3

VARIABLES MOST FREQUENTLY USED
IN ATACM1 AND ATACM2

Variable	Subroutine or COMMON	Description
ABAF(i,j,k)	INPUT	Fraction of type j aircraft on side k assigned to the ith mission which is vulnerable to airbase attack.
ALPHA	GAMES	Coefficient used in the linear interpolation algorithm to weight the objective function value corresponding to that point in the state space currently being examined.
ATTK	BATTLE	Number of ABA sorties flown against the opponent's airbase during the current one-cycle battle.
BETA(·,j)	INTERP	Weights used to linearly interpolate an objective function value for a point lying between two adjacent grid levels in dimension j. BETA(1,j) is the weight for the value corresponding to the lower level; BETA(2,j) the weight for the higher level.
BMIN(i)	GAMES	Minimum value in the ith row of the game matrix for the current state.
BOBJ	INIT TRIALS	Value of Blue's contribution to the objective function.
BTFP	BATTLE	Total ground firepower delivered by Blue during the current one-cycle battle.
CASF(j,k)	INPUT	Firepower per CAS sortie for an aircraft of type j on side k.
CASO(k)	BATTLE	Total CAS firepower delivered by side k during the current one-cycle battle.
CHECK	GAMES	Objective function value resulting from using specified Blue/Red strategies for one stage followed by the use of optimal conservative strategies by both sides for all subsequent stages. Used to compute MAXMIN/MINMAX bounds.
CHECKB	GAMES	Analogous to CHECK. Used to compute Blue's optimal MAXMIN play.
CHECKR	GAMES	Analogous to CHECK. Used to compute Red's optimal MINMAX play.
CNP(j,k)	BPARM	Current number of planes of type j on side k.

TABLE B-3 (Cont'd)

Variable	Subroutine or COMMON	Description
CNPV(j)	BATTLE	Current number of planes of type j vulnerable to the opponent's airbase attackers.
DELTA(j)	IPARM	Distance between adjacent "fine" grid levels in dimension j.
DFPRED(j,k)	INPUT	Division firepower reduction produced by a CAS sortie flown by aircraft type j on side k.
DIVFP(k)	INPUT	Firepower produced by a ground division of type k.
FEBA(i,j)	INPUT	ith coordinate of the jth point in the set of points which define FEBA movement as a function of Blue/Red force ratios.
FRACN	STRAT	Denominator of the minimum allocation fraction for the current aircraft type.
FRATIO	BATTLE	Ratio of Blue ground firepower to Red ground firepower for the current one-cycle battle.
IARRAY(·)	WORK	Work array used to store strategies, battle assessments, and MAXMIN/MINMAX plays and objective function values. EQUIVALENCED to XARRAY.
IB	BPARM	Index of the strategy employed by Blue in the current one-stage battle.
IBETA(·,j)	INTERP	Indices of grid levels lying on either side of an interpolation point in dimension j. IBETA(1,j) is one less than the subscript corresponding to the lower grid level; IBETA(2,j) is one less than the subscript corresponding to the higher.
IBH	INIT	High bound on the range of stages over which the current Blue strategy can be played.
IBIT(i)	IPARM	Rightmost bit in the ith 7-bit byte of the word used to store the results of a one-stage battle assessment. From left to right, bytes 1-4 are used to store levels of Blue aircraft of types 1-4; bytes 5-8 levels of Red aircraft of types 1-4.
IBL	INIT	Low bound on the range of stages over which the current Blue strategy can be played.
IBLURD(k)	WORKN	Hollerith constant equal to "BLUE" if k=1 or "RED" if k=2.

TABLE B-3 (Cont'd)

Variable	Subroutine or COMMON	Description
IBPLAY	GAMES	Index of Blue's optimal MAXMIN strategy.
IBR(k)	BPARM	Index of the strategy employed by side k in the current one-stage battle.
IDEM(·)	WORKN	Array of constants used to compute the number of the state corresponding to a set of grid-level indices. Constants are assigned in INIT.
IDINT(t,k)	WORKN	Latest stage for which the set of strategies available to side k during stage t are applicable.
IDUMMY(·)	READIN	Array used for temporary storage of input parameters prior to their being decoded.
IERR	ERROR	Flag indicating the severity of the last diagnostic message printed.
IETYPE(i)	ERR	Severity code associated with diagnostic i. Codes 1,2, and 3 correspond to "INFO", "ERROR", and "ABORT" respectively.
IGRID(i,j,k)	INPUT	ith grid level assigned to aircraft type j on side k.
IHEAD(·,·)	PRNTIN	Array of explanatory titles used to print input parameters in an easy-to-read format.
IKEY	READIN	Card key on the current input card.
IMAX	TRIALS	Rounded MINMAX bound on the value of the objective function for the current trial.
IMESSG(·,i)	ERR	Diagnostic message associated with error code i.
IMIN	TRIALS	Rounded MAXMIN bound on the value of the objective function for the current trial.
IMISS(i,j,k)	INPUT	Code of the ith mission assigned to aircraft type j on side k.
IMPNT(i,k)	WORKN	ith mission code assigned to aircraft types on side k.
INPNT(i,k)	WORKN	Index, within an aircraft type, of the ith mission code assigned to all aircraft types on side k.
INPUTZ(·)	INPUT	Single array EQUIVALENCED to COMMON block INPUT.
IOBJ	GAMES INIT	Blue's contribution to the value of the objective function resulting from a one-stage battle.

TABLE B-3 (Cont'd)

Variable	Subroutine or COMMON	Description
IOBJF	TRIALS	Rounded cumulative value of the objective function produced by playing optimal MAXMIN strategies against optimal MINMAX strategies during the current trial.
IOCBVB	WORKL	Beginning location of the area used in XARRAY to store current-stage MAXMIN values for Blue.
IOCBVR	WORKL	Beginning location of the area used in XARRAY to store current-stage MINMAX values for Red.
IOCEVB	WORKL	End location of the area used in XARRAY to store current-stage MAXMIN values for Blue.
IOCEVR	WORKL	End location of the area used in XARRAY to store current-stage MINMAX values for Red.
IOPTN(i)	TRIALS	Value of the ith print option specified on the current TRIAL card.
IPLAL(t,k)	TRIALS	Beginning location in XARRAY of the optimal allocation fractions used by side k during stage t of the current trial war.
IPNT(i)	SPARM	Index of the ith strategy for which an allocation fraction is not specified.
IPPNT(i)	TEMP	Pointer indicating the dimension in the state space which corresponds to the ith aircraft type assigned.
IPRINT(i)	INPUT	Value of the control parameter specified for the ith print option on the RUN card.
IR	BPARM	Index of the strategy employed by Red in the current one-stage battle.
IRA(·)	WORKN	Array of indices used in conjuction with the random access file TAPE2.
IRHI(i)	SINTVL	High bound on the range of Red strategies which can be played against the ith strategy of Blue.
IRLO(i)	SINTVL	Low bound on the range of Red strategies which can be played against the ith strategy of Blue.
IRPLAY	GAMES	Index of Red's optimal MINMAX strategy.
ISINT(i,k)	WORKN	Latest stage for which the ith strategy available to side k is applicable.

TABLE B-3 (Cont'd)

Variable	Subroutine or COMMON	Description
ISTAT	GAMES	Number of the state corresponding to a set of grid levels.
ITITLE(·)	INPUT	Array used to store the run title as specified on the RUN card.
ITP	READIN	Flag to indicate whether the parameters on the current card are for Blue or Red. ITP=1 corresponds to Blue, ITP=2 to Red.
ITPNT(i,k)	WORKN	Aircraft type to which the ith mission on side k is assigned.
ITYPE	SPARM	Aircraft type for which the current strategies are being generated.
IVERT(·,j)	GAMES	An NPTT-tuple of 1's and 2's representing the jth vertex of a "cube" in the state space. The number of possible tuples equals LPTT.
IWORD	GAMES INIT	Word containing both Blue and Red contributions to the value of the objective function resulting from a one-stage battle. Blue's contribution is packed into the leftmost 30 bits, Red's in the rightmost 30 bits.
JBETA(·,j,k)	IPARM	Indices of grid levels lying on either side of the kth "fine" grid level in dimension j. JBETA(1,j,k) is one less than the subscript corresponding to the lower grid level; JBETA(2,j,k) is one less than the subscript corresponding to the higher.
JDEX(·,·,k)	ROUND	Array of grid-level indices used to compute the state resulting from rounding the numbers of aircraft on side k down and the numbers opposing side k up.
JOBJ	GAMES INIT	Red's contribution to the value of the objective function resulting from a one-stage battle.
JOCBVB	WORKL	Beginning location of the area used in XARRAY to store next-stage MAXMIN values for Blue.
JOCBVR	WORKL	Beginning location of the area used in XARRAY to store next-stage MINMAX values for Red.
JOCEVB	WORKL	End location of the area used in XARRAY to store next-stage MAXMIN values for Blue.

TABLE B-3 (Cont'd)

Variable	Subroutine or COMMON	Description
JOCEVR	WORKL	End location of the area used in XARRAY to store next-stage MINMAX values for Red.
JOPTN(i)	TRIALS	Value of the ith print option specified on the previous TRIAL card.
JPRINT(i)	ATACM2	Value of the control parameter specified for the ith print option on the RUN card.
JRA(·)	WORKN	Array of indices used in conjunction with the random access file TAPE3.
JWORD	GAMES INIT	Word containing "fine" grid-level indices indicating the numbers of planes remaining after a one-stage battle.
KEVEL(·)	INIT	Array of "fine" grid-level indices indicating the numbers of planes remaining after a one-stage battle.
KEY(·)	READIN	Array of valid card keys recognized by READIN.
KOCBG	WORKL	Beginning location of the area used in IARRAY to store numbers of planes available after one-stage battle assessments.
KOCBVB	WORKL	Beginning location of the area used in XARRAY to store interpolated current-stage MAXMIN values for Blue.
KOCBVR	WORKL	Beginning location of the area used in XARRAY to store interpolated current-stage MINMAX values for Red.
KOCEG	WORKL	End location of the area used in IARRAY to store numbers of planes available after one-stage battle assessments.
KOCEVB	WORKL	End location of the area used in XARRAY to store interpolated current-stage MAXMIN values for Blue.
KOCEVR	WORKL	End location of the area used in XARRAY to store interpolated current-stage MINMAX values for Red.
LASTP	INPUT	Last stage thru which the current STRT card is applicable.
LEVEL	GAMES INIT	Index of the "fine" grid-level corresponding to the numbers of planes remaining after a one-stage battle.

TABLE B-3 (Cont'd)

Variable	Subroutine or COMMON	Description
LOCBG	WORKL	Beginning location of the area used in IARRAY to store objective function values resulting from one-stage battle assessments.
LOCBPB	WORKL	Beginning location of the area used in IARRAY to store the indices of optimal MAXMIN plays for Blue.
LOCBPR	WORKL	Beginning location of the area used in IARRAY to store the indices of optimal MINMAX plays for Red.
LOCBVB	WORKL	Beginning location of the area used in XARRAY to store interpolated next-stage MAXMIN values for Blue.
LOCBVR	WORKL	Beginning location of the area used in XARRAY to store interpolated next-stage MINMAX values for Red.
LOCEG	WORKL	End location of the area used in IARRAY to store objective function values resulting from one-stage battle assessments.
LOCEPB	WORKL	End location of the area used in IARRAY to store the indices of optimal MAXMIN plays for Blue.
LOCEPR	WORKL	End location of the area used in IARRAY to store indices of optimal MINMAX plays for Red.
LOCEVB	WORKL	End location of the area used in XARRAY to store interpolated next-stage MAXMIN values for Blue.
LOCEVR	WORKL	End location of the area used in XARRAY to store interpolated next-stage MINMAX values for Red.
LOCST(i,k)	WORKL	Beginning location in XARRAY of the allocation fractions which define the ith strategy of side k.
LOCWD	INIT	Location within IARRAY where the results of the next one-stage battle will be stored.
LPTT	GAMES	Total number of vertices on a "cube" in the state space. Equals 2 raised to the NPTT power.
LUNI	WORKN	Logical unit number of the primary input device (card reader). In the current version LUNI=5.
LUNO	WORKN	Logical unit number of the primary output device (line printer). In the current version LUNO=6.

TABLE B-3 (Cont'd)

Variable	Subroutine or COMMON	Description
MALOC(·)	INPUT	Input array used for temporary storage of STRT card parameters.
MFRAC	SPARM	Numerator of the fraction of aircraft unspecified on the current STRT card for the current aircraft type.
MISSN	STRAT	Number of missions assigned to the current aircraft type.
MMISS	SPARM	Number of missions assigned to the current aircraft type for which allocation fractions are not specified.
MSLOC	GAMES TRIALS	Number of the stage for which current calculations are being made.
MSTAGE	TRIALS	Number of stages specified on the current TRIAL card.
MSTOR(i)	STRAT	Numerator of the allocation fraction for the ith mission assigned to the current aircraft type.
MVEC(·)	ROUND	Array of grid-level indices used to compute the number of the corresponding state.
NALOC(i,j)	INPUT	Numerator of the allocation fraction specified on the current STRT card for the ith mission assigned to aircraft type j.
NBATLS	TIMER	Total number of battle evaluations required for each stage, state, and cycle.
NDAPST	INPUT	Number of cycles per stage.
NDIV(k)	INPUT	Number of ground divisions specified for side k.
NFRAC(j,k)	INPUT	Denominator of the minimum allocation fraction specified for aircraft type j on side k.
NFSAM(k)	INPUT	Number of forward SAMs specified for side k.
NFULST(k)	WORKN	Total number of strategies available to side k.
NGRID(j,k)	WORKN	Number of grid-levels specified for aircraft type j on side k.
NINGAM	WORKN GAMES	Number of elements in each one-stage/one-state game matrix.
NKEYS	READIN	Number of valid card keys recognized by READIN.

TABLE B-3 (Cont'd)

Variable	Subroutine or COMMON	Description
NMISS(j,k)	WORKN	Number of missions assigned to aircraft type j on side k.
NMISST(k)	WORKN	Total number of missions assigned to all aircraft types on side k.
NPNTS	TIMER	Total number of vertices on a "cube" in the state space. Equals 2 raised to the NTP power.
NPTT	GAMES	Total number of aircraft types assigned to Blue and Red.
NRSAM(k)	INPUT	Number of rear SAMs specified for side k.
NSHL(k)	INPUT	Number of aircraft shelters assigned to side k.
NSKIP	TRIALS	Number of records to be skipped on TAPE7 and TAPE8 before reading optimal plays and values for the first stage of the current trial war.
NSLOC	GAMES TRIALS	Number of the next stage for which calculations will be made. Equals MSLOC + 1.
NSTAGE	INPUT	Number of stages in the campaign.
NSTAT	WORKN	Total number of states for which optimal plays and objective function values are explicitly computed.
NSTOR(i)	SPARM	Numerator of the allocation fraction specified for the ith mission assigned to the current aircraft type.
NSTRAT(k)	WORKN	Total number of strategies available to side k.
NSTRTC(k)	WORKN	Number of STRT cards submitted for side k.
NTARG	BATTLE	Total number of planes vulnerable to the opponent's airbase attackers during the current one-cycle battle.
NTP	TIMER	Total number of aircraft types assigned to Blue and Red.
NTRIAL	TRIALS	Number of the current trial.
NTYPE(k)	WORKN	Number of aircraft types specified for side k.
NVEC(·)	ROUND	Array of grid-level indices used to compute the number of the corresponding state.
NWORK	WORKN	Length, in words, of the array XARRAY.

TABLE B-3 (Cont'd)

Variable	Subroutine or COMMON	Description
OBJEC(k,i)	BPARM	Contribution of side k to the value of the ith objective function f_i (see Equation A-1).
OBJF	TRIALS	Value of the objective function for the current stage produced by playing the optimal MAXMIN strategy against the optimal MINMAX strategy.
OWGHT(i)	INPUT	Weight w_i specified on the OWGHT card to be used as a multiplier on f_i (see Equation A-1).
PKAA(i,j,k)	INPUT	Probability an airbase attacker of type j on side k is killed by an opposing airbase defender of type i.
PKAAFS(j,k)	INPUT	Probability an airbase attacker of type j on side k is killed by an opposing forward SAM.
PKAARS(j,k)	INPUT	Probability an airbase attacker of type j on side k is killed by an opposing rear SAM.
PKAD(i,j,k)	INPUT	Probability an airbase defender of type j on side k is killed by an opposing airbase attacker of type i.
PKADES(i,j,k)	INPUT	Probability an airbase defender of type j on side k is killed by an opposing airbase attack escort of type i.
PKAEFS(j,k)	INPUT	Probability an airbase attack escort of type j on side k is killed by an opposing forward SAM.
PKAERS(j,k)	INPUT	Probability an airbase attack escort of type j on side k is killed by an opposing rear SAM.
PKBA(i,j,k)	INPUT	Probability a battlefield attacker of type j on side k is killed by an opposing battlefield defender of type i.
PKBAFS(j,k)	INPUT	Probability a battlefield attacker of type j on side k is killed by an opposing forward SAM.
PKBD(i,j,k)	INPUT	Probability a battlefield defender of type j on side k is killed by an opposing battlefield attacker of type i.
PKBDES(i,j,k)	INPUT	Probability a battlefield defender of type j on side k is killed by an opposing battlefield attack escort of type i.
PKBEFS(j,k)	INPUT	Probability a battlefield attack escort of type j on side k is killed by an opposing forward SAM.

TABLE B-3 (Cont'd)

Variable	Subroutine or COMMON	Description
PKESAD(i,j,k)	INPUT	Probability an airbase attack escort of type j on side k is killed by an opposing airbase defender of type i.
PKESBD(i,j,k)	INPUT	Probability a battlefield attack escort of type j on side k is killed by an opposing battlefield defender of type i.
PKFAFS(j,k)	INPUT	Probability a forward SAM attacker of type j on side k is killed by an opposing forward SAM.
PKFS(i,k)	INPUT	Probability a forward SAM on side k is killed by an opposing forward SAM attacker of type i.
PKNS(i,k)	INPUT	Probability a vulnerable, non-sheltered aircraft on side k is killed by an opposing airbase attacker of type i.
PKRAFS(j,k)	INPUT	Probability a rear SAM attacker of type j on side k is killed by an opposing forward SAM.
PKRARS(j,k)	INPUT	Probability a rear SAM attacker of type j on side k is killed by an opposing rear SAM.
PKRS(i,k)	INPUT	Probability a rear SAM on side k is killed by an opposing rear SAM attacker of type i.
PKSH(i,k)	INPUT	Probability a vulnerable, sheltered aircraft on side k is killed by an opposing airbase attacker of type i.
REIN(j,k,t)	INPUT	Number of reinforcement planes of type j on side k introduced at the beginning of stage t.
REINF(j,k,t)	INPUT	1.0 plus the fraction of reinforcement planes of type j on side k introduced at the beginning of stage t.
REMP(j,k)	BATTLE	Number of remaining planes of type j on side k after attrition.
RFSAM(k)	BATTLE	Number of undamaged forward SAMs on side k during the current one-cycle battle.
RMAX(j)	GAMES	Maximum value in the jth column of the game matrix for the current state.
ROBJ	INIT TRIALS	Value of Red's contribution to the objective function.
RRSAM(k)	BATTLE	Number of undamaged rear SAMs on side k during the current one-cycle battle.

TABLE B-3 (Cont'd)

Variable	Subroutine or COMMON	Description
RTFP	BATTLE	Total ground firepower delivered by Red during the current one-cycle battle.
STRATS(i,k,j)	WORK	Allocation fraction in strategy k for the ith mission assigned to aircraft type j. STRATS makes temporary use of part of the area occupied by XARRAY.
TARG	BATTLE	Total number of planes vulnerable to the opponent's airbase attackers during the current one-cycle battle.
TARGN	BATTLE	Number of unsheltered planes vulnerable to the opponent's airbase attackers during the current one-cycle battle.
TARGS	BATTLE	Number of sheltered planes vulnerable to the opponent's airbase attackers during the current one-cycle battle.
TCASO(k)	BATTLE	Total CAS firepower delivered by side k during the current one-stage battle.
TFIRE(k)	BATTLE	Total ground firepower delivered by side k during the current one-cycle battle.
TIMEC(k)	TIMER	Estimated CPU time required for the ith phase of calculations performed by ATACM1. Phases 1,2, and 3 are SETUP, BATTLES, and GAMES respectively.
TIMEI(i)	TIMER	Estimated I/O time required for the ith phase of calculations performed by ATACM1. Phases 1,2, and 3 are SETUP, BATTLES, and GAMES respectively.
TMOVE	BATTLE	Total FEBA movement during the current one-stage battle.
TNNK	BATTLE	Total number of unsheltered aircraft not killed by opposing airbase attackers during the current one-cycle battle.
TNP(i,k)	BATTLE	Total number of sorties assigned to be flown by planes prosecuting mission i for side k during the one-cycle battle.
TOBJF	TRIALS	Cumulative value of the objective function produced by playing optimal MAXMIN strategies against optimal MINMAX strategies during the current trial.

TABLE B-3 (Cont'd)

Variable	Subroutine or COMMON	Description
TOTFP(k)	BATTLE	Total firepower delivered by side k during the current one-stage battle.
TSNK	BATTLE	Total number of sheltered aircraft not killed by opposing airbase attackers during the current one-cycle battle.
TXOBJF(i)	TRIALS	Cumulative value of objective function i produced by playing the optimal MAXMIN strategy against the optimal MINMAX strategy during the current trial.
VALU(j,k)	INPUT	Residual value of an undamaged plane of type j available to side k at the end of the war.
WGHT(t,k)	INPUT	Value of weight b_t if k=1 or r_t if k=2 as specified on the WGHT card (see Equations A-3 thru A-5).
WORKLZ(·)	WORKL	Single array EQUIVALENCED to COMMON block WORKL.
WORKNZ(·)	WORKN	Single array EQUIVALENCED to COMMON block WORKN.
WORKZ(·)	WORK	Single array EQUIVALENCED to COMMON block WORK.
XARRAY(·)	WORK	Work array used to store strategies, battle assessments, and MAXMIN/MINMAX plays and objective function values. EQUIVALENCED to IARRAY.
XATT	BATTLE	Number of attack sorties during the current phase of the one-cycle battle.
XBETA(·,j,k)	IPARM	Weights used to linearly interpolate an objective function value for the kth "fine" grid-level lying between two adjacent grid-levels in dimension j. XBETA(1,j,k) is the weight for the value corresponding to the lower level, XBETA(2,j,k) the weight for the higher level.
XFSAM	BATTLE	Number of forward SAMs available to the defending side during the current one-cycle battle.
XGRID(i,j,k)	INPUT	ith grid-level assigned to aircraft type j on side k.
XMOVE	BATTLE	FEBA movement during the current one-cycle battle.

TABLE B-3 (Cont'd)

Variable	Subroutine or COMMON	Description
XNENG	BATTLE	Number of one-on-one engagements between attackers and opponents during the current phase of a one-cycle battle.
XNP(i,j,k)	BPARM	Number of successful sorties flown by planes of type j prosecuting mission i for side k during the current one-cycle battle.
XOBJF(i,t)	TRIALS	Value of objective function i produced by playing the optimal MAXMIN strategy against the optimal MINMAX strategy during stage t.
XOPP	BATTLE	Number of sorties opposing the attackers during the current phase of the one-cycle battle.
XRSAM	BATTLE	Number of rear SAMs available to the defending side during the current one-cycle battle.
XSORT(i,j,k)	INPUT	Sortie rate for aircraft of type j on side k assigned to the ith mission.

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER PAB - 249 ✓	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) 6 ATACM: ACDA Tactical Air Campaign Model.		5. TYPE OF REPORT & PERIOD COVERED 9 Final Rep[redacted]t.
		6. PERFORMING ORG. REPORT NUMBER 14 KFR [redacted]44-75 ✓
7. AUTHOR(s) 10 John R./Fish		8. CONTRACT OR GRANT NUMBER(s) 15 ACDA/PAB-249
9. PERFORMING ORGANIZATION NAME AND ADDRESS KETRON, Inc. ✓ 1400 Wilson Blvd. Arlington, Va. 22209		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 12 161p.
11. CONTROLLING OFFICE NAME AND ADDRESS U.S. Arms Control and Disarmament Agency 21st & Virginia Ave's., N.W. Washington, D.C. 20451		12. REPORT DATE 11 Oct[redacted]75
		13. NUMBER OF PAGES 158
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION DOWNGRADING SCHEDULE

~~TdE~~ 21 JAN 1976

16. DISTRIBUTION STATEMENT (of this Report)

Distribution limited to U.S. Government agencies only. Other requests for this document must be referred to Director, U.S. Arms Control and Disarmament Agency.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Tactical air war	Multi-stage game
Aircraft allocation to missions	Dynamic programming
Force mix	Computer simulation
	Optimization

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

ATACM is a computer model designed and built for the Arms Control and Disarmament Agency for use in analyzing the impact of various force mixes upon a tactical airwar in Europe between NATO and Warsaw Pact forces. ATACM models an air campaign as a zero-sum staged game and employs dynamic programming to solve this game for approximate, optimal MAXMIN/MINMAX aircraft allocation strategies for the opposing sides at each stage of the

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

390 861 Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

VB

#20 (continued)

campaign. The model permits multiple aircraft types with user-assigned missions, numerical and fractional reinforcements as a function of stage, and user selection of the objective function used to generate the optimal strategies.

Descriptions of the problem formulation and the engagement and optimization methodologies used to solve it are presented along with a user's guide and CDC 6600 FORTRAN listings.